



NON-LINEAR OPTIMIZATION APPLIED TO  
ANGLE-OF-ARRIVAL SATELLITE-BASED  
GEOLOCATION WITH CORRELATED  
MEASUREMENTS

THESIS

Joshua S. Sprang, 2d Lt, USAF  
AFIT-ENG-MS-15-M-044

DEPARTMENT OF THE AIR FORCE  
AIR UNIVERSITY

***AIR FORCE INSTITUTE OF TECHNOLOGY***

---

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A  
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views expressed in this document are those of the author and do not reflect the official policy or position of the United States Air Force, the United States Department of Defense or the United States Government. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT-ENG-MS-15-M-044

NON-LINEAR OPTIMIZATION APPLIED TO ANGLE-OF-ARRIVAL  
SATELLITE-BASED GEOLOCATION WITH CORRELATED  
MEASUREMENTS

THESIS

Presented to the Faculty  
Department of Electrical and Computer Engineering  
Graduate School of Engineering and Management  
Air Force Institute of Technology  
Air University  
Air Education and Training Command  
in Partial Fulfillment of the Requirements for the  
Degree of Master of Science in Electrical Engineering

Joshua S. Sprang, B.S.E.E., B.S.Cp.E  
2d Lt, USAF

March 2015

DISTRIBUTION STATEMENT A  
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AFIT-ENG-MS-15-M-044

NON-LINEAR OPTIMIZATION APPLIED TO ANGLE-OF-ARRIVAL  
SATELLITE-BASED GEOLOCATION WITH CORRELATED  
MEASUREMENTS

THESIS

Joshua S. Sprang, B.S.E.E., B.S.Cp.E  
2d Lt, USAF

Committee Membership:

Dr. A. Terzuoli  
Chair

Dr. C. Taylor  
Member

Col M. Sambora, PhD  
Member

## Abstract

A common remote sensing application is producing geolocation estimates for an object of interest from multiple sensor platforms. Geolocation estimates are desired to help improve situational awareness when dealing with space objects that do not actively broadcast their location. A depiction of the error parameters are calculated in conjunction with the positional estimates. Satellite based remote sensors often use time of arrival (TDoA) or received signal strength (RSS) measurements in order to perform the geolocation task. This paper presents an optimization algorithm for multiple non-time coincident angle of arrival (AoA) measurements from different sensor platforms. Problems occur when multiple measurements from a single sensor are used to estimate a location due to correlations in sensor error. A non-linear optimization approach is presented for determining geolocation estimates and their associated error parameters. The error parameters directly reflect the error present on the individual measurements used to produce the position estimates. Correlations in errors are dealt with by augmenting the non-linear optimization with a covariance intersection algorithm. Finally, the ability to account for correlated errors within the optimization algorithm is analyzed using Monte-Carlo simulations. The ability to describe an objects location with a given confidence helps aid in the analysis of the system at large.

# Table of Contents

	Page
Abstract .....	iv
List of Figures .....	vii
List of Symbols .....	x
I. Introduction .....	1
1.1 Introduction .....	1
1.2 Background .....	2
Localization .....	2
Sensors .....	4
Optimization .....	5
1.3 Problem Statement .....	5
1.4 Methodology .....	7
1.5 Conclusion .....	8
II. Literature .....	10
2.1 Introduction .....	10
2.2 Estimation using Optimization .....	10
2.3 Error Calculation .....	14
2.4 Geolocation Algorithms .....	17
2.5 Bias .....	22
2.6 Covariance Intersection .....	24
2.7 Path Optimization .....	27
2.8 Analysis .....	28
2.9 Conclusion .....	31
III. Methodology .....	32
3.1 Introduction .....	32
3.2 Theory .....	33
Acceleration Estimation .....	33
Covariance Intersection for Non-Linear Optimization .....	35
Transition Matrices .....	40
3.3 Materials and Equipment .....	42
3.4 Procedures and Process .....	45
3.5 Conclusion .....	49

	Page
IV. Results .....	52
4.1 Introduction .....	52
4.2 Results .....	53
4.3 Conclusion .....	67
V. Conclusion .....	70
5.1 Introduction .....	70
5.2 Simulations .....	70
5.3 Conclusion .....	73
Future Work .....	73
Bibliography .....	78

## List of Figures

Figure		Page
1.	Earth centered earth fixed coordinate system [1] .....	3
2.	The azimuth $\theta$ and elevation $\phi$ angles that constitute an angle of arrival measurement .....	6
3.	Resulting solutions for an object's position due to line of sight error .....	11
4.	Depiction of a single iteration of Newton's Method .....	12
5.	Depiction of the triangulation algorithm from line of sight vector measurements .....	18
6.	Visualization of the static non-linear optimization assumptions .....	19
7.	Visualization of the velocity non-linear optimization assumptions .....	20
8.	Visualization of acceleration and velocity non-linear optimization assumptions .....	21
9.	Truncation of signal due to analog-to-digital quantization .....	23
10.	Probabilistic effects on error due to analog-to-digital quantization .....	24
11.	Possible covariance ellipsoids (left) and the corresponding covariance ellipse using Covariance Intersection (CI) (right) .....	25
12.	Resulting covariance ellipsoids (black) from two angle of arrival measurements with depicted error (blue) on an angle of arrival measurement (red) .....	26
13.	Resulting covariance ellipsoids (black) from two angle of arrival measurements with depicted error (blue) on an angle of arrival measurement (red) when true covariance ellipse has different amounts of correlations. ....	27
14.	Depiction of the transition matrix (red) compared to position estimates based solely on measurements (blue) .....	28

Figure	Page
15.	Object Path simulated using differential equations ..... 43
16.	Bias simulation with added random noise from bias and noise with no measurements present ..... 45
17.	Monte-Carlo simulation without simulating bias error ..... 46
18.	Covariance ellipsoids along a path for the velocity non-linear optimization and velocity non-linear optimization using Covariance Intersection ..... 47
19.	True object paths (blue) and estimation paths (red) of the velocity non-linear optimization and velocity non-linear optimization using Covariance Intersection with associated total calculated error for the two different paths in meters ..... 48
20.	Monte-Carlo simulation of Static Non-Linear Optimization used on a circular path of an object ..... 53
21.	Monte-Carlo simulation of Velocity Non-Linear Optimization used on a circular object path ..... 54
22.	Monte-Carlo simulation of Velocity Non-Linear Optimization with Covariance Intersection used on a circular object path ..... 55
23.	Normalized estimation error squared (NEES) values for the different algorithms on a circular path ..... 55
24.	Mean squared error (MSE) for the different algorithms on a circular path ..... 56
25.	Average ellipsoid sizes for the different algorithms on a circular path ..... 57
26.	Monte-Carlo simulation of the Static Non-Linear Optimization used on a complex object path ..... 58
27.	Monte-Carlo simulation of the velocity non-linear optimization used on a complex object path ..... 58
28.	Monte-Carlo simulation of the Velocity Non-Linear Optimization with Covariance Intersection used on a complex object path ..... 59

Figure	Page
29.	NEES values for the different algorithms on a staged path . . . . . 60
30.	MSE for the different algorithms on a staged path . . . . . 61
31.	Average ellipsoid sizes for the different algorithms on a staged path . . . . . 62
32.	Time variance testing on the different algorithms for a circular path . . . . . 63
33.	Residual testing on the different algorithms for a circular path . . . . . 63
34.	Speed testing on the different algorithms for a circular path . . . . . 64
35.	Window testing on the different algorithms for a circular path . . . . . 65
36.	Windowing testing on the different algorithms for a complex path . . . . . 66
37.	NEES values for multiple estimation points occurring over time on a circular path . . . . . 68
38.	NEES values for multiple estimation points occurring over time on a staged path . . . . . 69
39.	Flowchart of Non-Linear Optimization (NLO) algorithm incorporating Expectation Maximization . . . . . 75
40.	Initial test of an Expectation Maximization algorithm on estimated acceleration values from a single path . . . . . 76
41.	Path of object used in initial Expectation Maximization test . . . . . 77

## List of Symbols

$\mathbf{x}$	
$f(x)$	Function of $x$
$x_0$	Initial estimate
$x_1$	Estimate 1
$\Omega$	Measurement vector
$\mathbf{X}$	State vector
$\mathbf{P}$	Object position vector
$\mathbf{V}$	Object velocity vector
$\mathbf{A}$	Object acceleration vector
$\mathbf{J}$	Jacobian matrix
$\mathbf{J}_{trans}$	Transition matrix concatenated to the Jacobian matrix
$w$	General weight variable
$\Sigma_{\Omega}$	Measurement covariance matrix
$\Sigma_X$	State covariance matrix
$\Sigma_{trans}$	Covariance matrix corresponding to the transisiton matrix
$\Sigma_c$	General covariance intersection covariance matrix
$\Sigma_1$	Measurement 1's covariance matrix
$\mathbf{I}(\mathbf{X})$	Fisher information matrix
$g$	Confidence region
$\mathbf{L}$	Lower triangular matrix from Cholesky decomposition
$\mathcal{S}$	Vector of sphere vertices
$S$	Sensor position vector
$\mathbf{U}, \mathbf{S}, \mathbf{B}$	Resulting matrices from singular value decomposition
$\sigma$	Scalar standard deviation value

$d$	Distance from estimated position to a line of sight vector
$D$	Total squared distance from estimated position to all LOS vectors
$\theta$	Azimuth angle
$\phi$	Elevation angle
$x, y, z$	Cartesian position components
$\dot{x}, \dot{y}, \dot{z}$	Cartesian velocity components
$\ddot{x}, \ddot{y}, \ddot{z}$	Cartesian acceleration components
$t$	Time
$\tau$	Propagation time delay
$\epsilon$	Error
$V$	Voltage
$\mathcal{U}$	Uniform distribution
$\chi_3^2$	Chi-squared distribution with three degrees of freedom
$\bar{\mathbf{m}}_c$	General covariance intersection mean
$\bar{\mathbf{m}}_1$	Measurement 1 mean
$\omega$	Covariance intersection weight
$\mathbf{W}$	Covariance intersection weighting matrix
$\epsilon_{nees}$	Normalized estimation error squared
$n_X$	Degrees of freedom for state $\mathbf{X}$
$\mathbf{X}_{stat}$	State vector for static non-linear optimization
$\mathbf{X}_{vnlo}$	State vector for velocity non-linear optimization
$h$	Small value for approximating derivatives
$O(h)$	Order of magnitude of $h$
$M$	Number of rows or columns in a matrix
$N$	Number of rows or columns in a matrix
$K$	Number of rows or columns in a matrix

$m$	Specific row or column in a matrix
$n$	Specific row or column in a matrix
$k$	Specific row or column in a matrix
$i, j$	Terms within a summation
$\mathbf{A}, \mathbf{B}, \mathbf{C}$	General matrices
$a, b, c$	Single cells within a matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ respectively

# NON-LINEAR OPTIMIZATION APPLIED TO ANGLE-OF-ARRIVAL SATELLITE-BASED GEOLOCATION WITH CORRELATED MEASUREMENTS

## I. Introduction

### 1.1 Introduction

Military object localization often requires the use of multiple sensors to create a single point estimate. Multiple point estimates in succession create an object's location. The act of determining an object's path using external sensors with respect to the earth is called geolocation. An object can determine its location through the use of the global positioning system (GPS) or other means of positioning including but not limited to the use of inertial measurement units (IMUs), magnetometers, or digital compasses [2]. However, geolocation from remote sensors is desired when an object is unable to relay its position to a controller. Objects in this category include deactivated satellites, meteors, and general space debris. The information in this chapter will discuss the motivation and background of the geolocation algorithm with the scope of the applications for this research. The main topics of discussion are object geolocation and the error depiction of an object's location. A system of sensors being used to create an estimate can be accurately gauged and the certainty in a localized position can be inferred when the error produced in the simulations that is associated with an estimated object location is known. Therefore, current sensor systems can be analyzed for any deficiencies in coverage, and a more complete description of the objects within the space environment can be realized.

## 1.2 Background

In this section, the topics of localization, coordinate systems and the fusion of data between coordinate systems, remote sensing, and the optimization process are discussed.

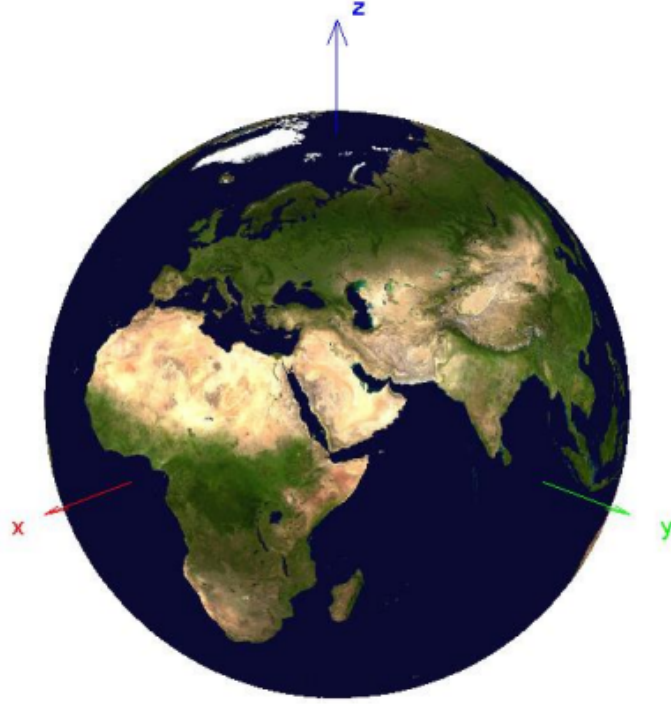
### **Localization.**

Localization refers to the practice of finding the local area that an object resides in. The idea is synonymous to determining an object's location; however, localization more specifically refers to the determination of an entire object state. A state estimate is composed of some combination of position, velocity, and acceleration of an object. The purpose of localization is to describe an object using a set of given measurements. Therefore a good localization algorithm is able to produce the best description of an object given the information from a set of measurements.

Determining a position requires the use of a coordinate system. Every coordinate requires a point of origin and a combination of distance and angular quantities to determine the position with reference to the defined origin. Within three dimensions, the most common coordinate systems are Cartesian, Cylindrical, and Spherical. The variables describing the position using the three common coordinate systems are sets of magnitudes and angles. In a Cartesian coordinate system, the three variables describing a position are vector magnitudes, while a Spherical coordinate system uses two angles and a magnitude. It is possible to translate a measurement from one coordinate system into another coordinate system thus allowing for the ability to fuse multiple measurements from different sensors into a single coordinate system.

When performing geolocalization, two major coordinate systems are the earth centered earth fixed (ECEF) and earth centered inertial (ECI) systems. This research will focus on the ECEF system, which uses Cartesian based coordinates [3]. This point

of origin in the ECEF coordinate system is the center of the Earth in space and not in mass. The  $x$ -axis passes through the Greenwich meridian, the  $z$ -axis points north and lies on the Earth's axis of rotation, and the  $y$ -axis completes the coordinate system by being orthogonal to the  $x$  and  $z$  axis forming the right-handed system. This system is depicted in Figure 1.



**Figure 1. Earth centered earth fixed coordinate system [1]**

Having the  $x$ -axis defined by the Greenwich meridian, a location on earth is uniquely defined by a given ECEF coordinate that does not change over time. This is not true for celestial coordinate systems that use the vernal equinox as one of the primary directions which causes a location's coordinate on Earth to change due to the Earth's spin. The ECEF coordinate system is important for any object that may end up entering Earth's atmosphere because the coordinate system describes where it will re-enter relative to Earth's surface.

## **Sensors.**

A sensor is any device that measures a physical quantity. To produce localization estimates, information is gathered about an object through the use of multiple sensor measurements. The more information gathered due to the increase in either quantity or quality of sensor measurements, the more accurate depiction of an object state can be formed.

For the purpose of geolocalization, sensors produce measurements describing an object's position and motion. A remote sensor produces a measurement at a distance from the object of interest. A measurement is created by measuring the strength of a signal received, or finding the direction at which a signal is emanating. The angle of arrival (AoA) is the direction of the signal relative to the sensor's reference. A measurement taken is with respect to a sensor's position; therefore, multiple sensors will produce data in multiple coordinate systems. Assuming a sensor's location is known, multiple measurements can be translated from their original coordinate systems to a single coordinate system and used to produce a single position estimate for the object of interest.

Sensors have inaccuracies influenced by different quantities of error. Components of a sensor and the environment of a sensor can cause errors in the measurements. A common assumption for error in different applications is that each instance of error in a measurement is unaffected by the previous instance of error and each instance of error is based on a random probability distribution. This assumption is known as independent and identically distributed (IID). If an instance of error is related to the previous instance, the error is correlated and thus no longer considered IID. When measurements contain error, there is no longer a unique solution for a calculated object state based on the measurements, requiring the use of an optimization algorithm.

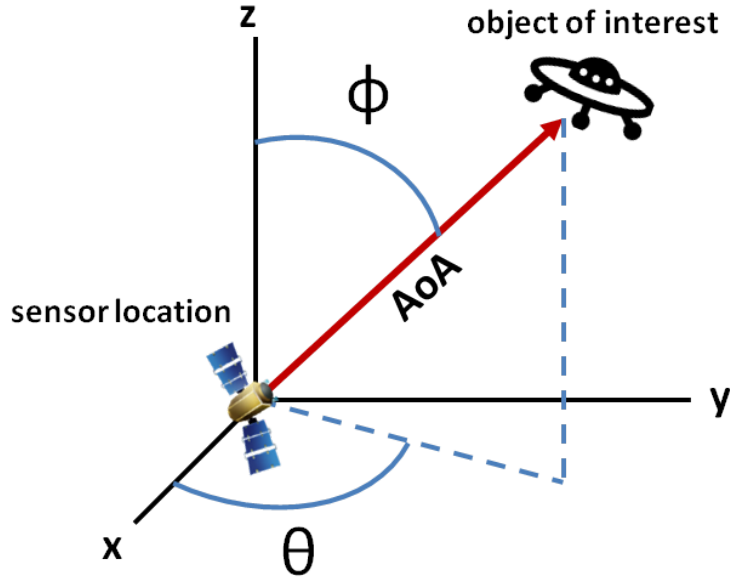
## Optimization.

One means of converging to an ideal solution given a set of measurements is optimization. Error in sensors causes measurements to contradict one another; therefore, a single solution to an object's location based on the measurements may not exist. Finding the solution that minimizes a cost function is defined as optimization [4]. Optimization generally requires finding either a local maximum or minimum to a set of equations. When performing geo-localization, the intent of the optimization is to find the minimum amount of error for an object's position. The minimum error is defined to be the estimate that best agrees with the measurements used to produce the estimate.

Optimization can be performed iteratively to converge to a solution using the gradients of the non-linear functions. For each iteration, the optimization technique produces a solution closer to that of the optimal solution. Different algorithms converge at different rates to the optimal solution. Examples of iterative optimization algorithms are gradient of descent and Newton's method. The iterative methods are useful compared to direct methods of optimization because they are able to simplify non-linear sets of equations.

### 1.3 Problem Statement

As stated in the introduction, this paper will concentrate on the localization from data produced by remote sensors. The data, assumptions, and process of geolocalization are presented in the following section. A measurement is captured passively by sensing an electromagnetic signal emanating from the object of interest. With an origin at the sensor's location, the measurement is a collaboration of an azimuth  $\theta$  and elevation  $\phi$  angle. The angles together constitute an AoA measurement as shown in Figure 2. Using non-time coincident AoA data from sensors, the research will



**Figure 2.** The azimuth  $\theta$  and elevation  $\phi$  angles that constitute an angle of arrival measurement

investigate the possibility of producing accurate error estimates based on the optimized geolocation estimate. AoA localization is commonly used in many localization practices. Wireless sensor networks often use AoA measurements to create an accurate map of each sensor node [5][6][7]. AoA measurements in the practice of wireless sensor positions require antenna arrays and thus more infrastructure than time difference of arrival (ToA) or received signal strength (RSS) measurements. ToA and RSS measurements are considered distance based measurements while AoA measurements are considered direction based measurements [8]. Direction based measurements are not as susceptible to the variabilities of propagation losses as distance measurements [7] counteracting the additional infrastructure costs. Variations in propagation losses can be due to changes in atmosphere, weather, or other particles and forces between the sensor and the object of interest. Therefore, more accurate localization estimations can be produced using AoA measurements, and error is better approximated. Geolocation using space based sensors is a common practice for a variety of ap-

plications. One practice is the localization of ships using ToA measurements [9]. The localization specifically uses the technique because the ships already relay a signal using the automatic identification system. By measuring the arrival times of the message transmissions, satellites are able to determine the ship's location when GPS signals are unable to be used. ToA measurements are common for many satellite based localization schemes including GPS. Research in refining the localization methods using ToA and RSS is continuing to this day [10] [11]. Because these passive measurements create a non-linear set of equations to solve, different numerical methods for solving the optimization problems are continually evolving. As computing becomes faster and more relied upon, the amount of processing that can be done on the signals becomes more comprehensive. Therefore, better estimates are able to be achieved.

## 1.4 Methodology

Prior research using the optimization techniques of interest assumed errors on sensors were IID. The over-simplified assumptions of error cause the estimated size of error to be overly optimistic. By increasing the complexity of the simulations of error on the sensors, more realistic analyses with respect to error representation on the localization algorithms will be performed.

The geolocation algorithm will be adjusted to take into consideration the higher complexity of the sensor error. The geolocation algorithm uses a weighted Gauss-Newton algorithm to produce the geo-location estimate and the calculated error. The weighting is derived from the expected amount of variation on the sensor's AoA measurements. Therefore, the geo-location algorithm is able to produce an expected error for the position estimation based on the individual sensors that were used to create the estimation.

The analysis of the geolocation algorithm is done using the MATLAB environment. The research takes into consideration how best to simulate the remote sensors along with the object being localized. Each simulation has some amount of randomness to accurately depict different real world scenarios.

Sensor simulations are set up such that each sensor is on-board an earth orbiting satellite. Satellite generation is based on the expected orbital mechanics. Therefore, the algorithm takes into consideration not only the dynamic object that is being located, but also the system of moving sensors. To thoroughly analyze the algorithm, the object of interest will be simulated in a variety of ways. The two main categories of paths are called complex and simple. A complex path has changes in the forces acting upon it over a given period of time. We predict the more complex a path is, the more error will be incurred on the estimates. This is because assumptions concerning the object of interest can break down when its path becomes too complex. An object of interest is also simulated using a less dynamic orbiting path. The path is considered simple because other than earth's gravitational pull, no other forces are acting upon the body. By creating different types of simulations, the localization algorithm is evaluated to ensure it is able to produce accurate estimates when analyzing any object it may encounter.

Monte-Carlo simulations are used to obtain statistical results on the ability for the algorithm to locate an object. Simulations allow for the ability to know the absolute truth of an object's position. Therefore, the simulations can be analyzed using the MSE and NEES.

## **1.5 Conclusion**

The purpose of the research is to determine the ability to estimate the position of an object while determining the confidence in each location estimate. The error in

the research is assumed to no longer be IID. Therefore, adjustments to the current geolocation algorithm are needed to produce accurate location estimates. This research will analyze how correlated error can be accounted for within the geolocation algorithm. The paper describes how to create a more realistic simulation of error using a bias on AoA measurements, the ability of the current NLO algorithm to localize under the new simulations, and how the new additions to the NLO algorithm account for the bias within the simulated error.

Chapter II discusses the background literature of the geolocation algorithm. Topics include simulating sources of error on sensors, the Gauss-Newton algorithm used for the NLO, the CI algorithm, and the use of transition matrices. Chapter III is the methodology section of the thesis. It describes how the simulations are formed, how the algorithms are formed, and how testing of the algorithms will occur. Chapter IV is the data and results section detailing the analysis of the data and results as described in Chapter III. Finally, Chapter V concludes the research based on the data produced in the previous Chapter, and concludes with future work for the research.

## II. Literature

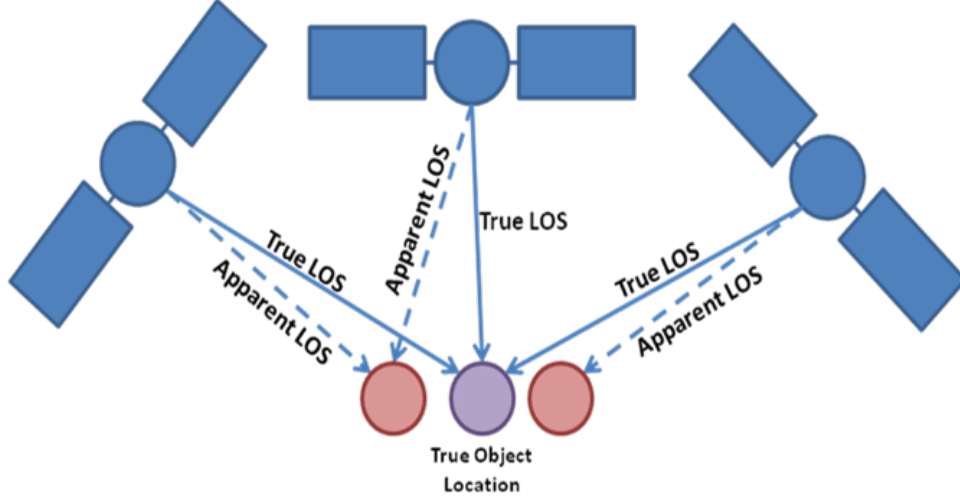
### 2.1 Introduction

The theoretical basis of the research in this paper is discussed in this chapter. Sensor measurements always have an error associated with them. The focus of the research is to create a localization algorithm that will take into account sensor uncertainty due to errors, and minimize the uncertainty in the estimated position. Localization is the estimation of an object of interest's position including tracking the object over a period of time [12]. Remote sensing refers to the gathering of information from a remote location. For this research, the remote sensing application refers to satellite sensor's line of sight (LOS) measurements in terms of their AoA. The research focuses on the mitigation of error correlations through mathematical processes and optimization. A review on the depiction of error concludes the chapter. Error depiction in conjunction with the localization is the final goal of the research, with the intent of creating a method for analyzing the ability for remote sensors to locate an object's position.

### 2.2 Estimation using Optimization

Optimization is the process of finding the optimal solution when multiple solutions to a problem are present. Multiple solutions to a problem occur when measurements have some amount of error induced on them. The error causes the measurements to contradict one another, thus making multiple solutions a possibility. A depiction of a possible contradiction between sensor measurements is shown in Figure 3.

With three different measurements being used to locate an object, when error is induced on the true LOS, multiple object locations become apparent. Finding an optimal solution from a set of measurements can be performed either directly or



**Figure 3. Resulting solutions for an object's position due to line of sight error**

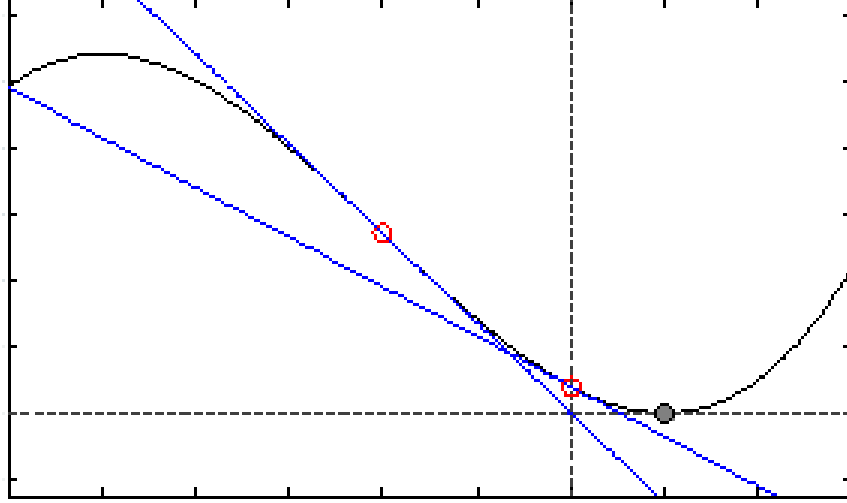
iteratively. Direct methods of optimization do not always have solutions for non-linear sets of equations. To account for this limitation, iterative optimization techniques are used simplifying the calculations of computationally complex systems of equations.

The research focuses on NLO as the base algorithm for localization [1]. NLO is based on Newton's method. In this method, a set of non-linear functions is simplified by a set of linear functions describing the non-linear functions at a given estimation point. A solution is estimated from the linear set of functions and that solution is used as the new estimate for another iteration of the method. A one-dimensional example of Newton's method is shown in Figure 4.

Starting from an initial estimate near the solution, Newton's method finds the next position from a tangential line for a function  $f$  laying on the initial estimation point. Therefore, the relationship between the next point in the estimation  $x_1$  and the initial estimation  $x_0$  can be described by the equation

$$f(x_1) - f(x_0) = \frac{df(x_0)}{dx}(x_1 - x_0). \quad (1)$$

Equation 1 is the point slope form of the tangent line at the initial estimate.



**Figure 4. Depiction of a single iteration of Newton's Method**

Modifying Newton's method to account for multiple non-linear functions results in the Gauss-Newton algorithm. The Gauss-Newton algorithm uses the Jacobian of measurements with respect to the values being estimated in place of the derivative of the function in the one dimensional case in equation 1. The Jacobian is used because multiple variables are being optimized within the sensor measurements. As with Newton's method, the Gauss-Newton algorithm starts out with an initial estimate for the optimal solution. The change from the initial estimate to the next best estimate is found on each iteration, and the calculated change corrects the estimate and the corrected estimate is used as the initial estimate in the next iteration of the algorithm. The equation for the Gauss-Newton algorithm uses  $\Delta\Omega$  as the change of the observations needed to produce the current position estimate from the initial observations,  $\Delta\mathbf{P}$  as the change of the new estimated positions from the initial position estimates, and  $\mathbf{J}(\mathbf{P}_{init})$  as the Jacobian matrix at the initial position estimate producing

$$\Delta\Omega = \mathbf{J}(\mathbf{P}_{init})\Delta\mathbf{P}. \quad (2)$$

The Jacobian in equation 2 is a matrix of partial derivatives with respect to the variables calculated in the state matrix. Assuming only the position is being estimated, the variables in the state matrix are the ECEF coordinates of the object of interest  $x$ ,  $y$ , and  $z$ . Therefore, the functions being differentiated producing the partial derivatives are the equations for the azimuth and elevation angles from the variables  $x$ ,  $y$ , and  $z$  via

$$\mathbf{\Omega}(\mathbf{P}) = [\theta(\mathbf{P}), \phi(\mathbf{P})]^\top = \left[ \tan^{-1} \left( \frac{y}{x} \right), \tan^{-1} \left( \frac{\sqrt{x^2 + y^2}}{z} \right) \right]^\top. \quad (3)$$

The Jacobian is a  $2M \times 3$  matrix where  $M$  is the number of AoA measurements used to produce a single state estimate.

The optimal estimate for the given measurements is produced from the Gauss-Newton algorithm. With each iteration of the Gauss-Newton algorithm, the estimate converges the estimate on the true optimum for the given measurements. The amount of change from one iteration to the next is given by  $\Delta \mathbf{P}$ . To solve for  $\Delta \mathbf{P}$  from equation 2, a weighted least mean squares of the Gauss-Newton algorithm using a weighting factor  $w$  and measurement covariance matrix  $\mathbf{\Sigma}_\Omega$  is applied using

$$\begin{aligned} w\mathbf{J}(\mathbf{P}_{init})\Delta\mathbf{P} &= w\Delta\mathbf{\Omega}, \\ \mathbf{J}^\top w^\top w\mathbf{J}\Delta\mathbf{P} &= \mathbf{J}^\top w^\top w\Delta\mathbf{\Omega}, \\ \Delta\mathbf{P} &= (\mathbf{J}^\top w^\top w\mathbf{J})^{-1} \mathbf{J}^\top w^\top w\Delta\mathbf{\Omega}, \\ \Delta\mathbf{P} &= (\mathbf{J}^\top \mathbf{\Sigma}_\Omega^{-1} \mathbf{J})^{-1} \mathbf{J}^\top \mathbf{\Sigma}_\Omega^{-1} \Delta\mathbf{\Omega}. \end{aligned} \quad (4)$$

$w$  is a weighting factor that when multiplied by the transpose of itself defines the inverse covariance matrix of the sensor AoA measurements  $\mathbf{\Sigma}_\Omega^{-1}$ . The Jacobian term  $\mathbf{J}$  is based on the AoA terms from the previous iteration, and referencing Figure 4, these terms will eventually converge on an optimal estimate when an accurate initial

estimate is used.

Let us consider an example where the NLO is being applied to two measurements of azimuth and elevation from a single sensor. The  $\mathbf{J}$  and  $\Sigma_{\Omega}^{-1}$  matrices for this example are defined as

$$\Sigma_{\Omega}^{-1} = \begin{bmatrix} \frac{1}{Var(\theta_1)} & 0 & 0 & 0 \\ 0 & \frac{1}{Var(\phi_1)} & 0 & 0 \\ 0 & 0 & \frac{1}{Var(\theta_2)} & 0 \\ 0 & 0 & 0 & \frac{1}{Var(\phi_2)} \end{bmatrix},$$

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \theta_1}{\partial x} & \frac{\partial \theta_1}{\partial y} & \frac{\partial \theta_1}{\partial z} \\ \frac{\partial \phi_1}{\partial x} & \frac{\partial \phi_1}{\partial y} & \frac{\partial \phi_1}{\partial z} \\ \frac{\partial \theta_2}{\partial x} & \frac{\partial \theta_2}{\partial y} & \frac{\partial \theta_2}{\partial z} \\ \frac{\partial \phi_2}{\partial x} & \frac{\partial \phi_2}{\partial y} & \frac{\partial \phi_2}{\partial z} \end{bmatrix}. \quad (5)$$

Each measurement as stated previously, adds two rows to the  $\mathbf{J}$ .

### 2.3 Error Calculation

Localization uses a given set of data to acquire a state estimate. We assume the given data has determined whether an object is present or not; therefore, the NLO needs to estimate the position along with any other attributes for the object of interest from the data. Once an estimate is found, we analyze whether the NLO is a good estimator for the given data using the Cramer-Rao lower bound (CRLB).

The CRLB gives the lower bound on the variance of an unbiased estimator given the data used [13]. Using a set of data  $\Omega$ , and estimating a state  $\mathbf{X}$ , the estimator first needs to satisfy the condition

$$E \left[ \frac{\partial \ln(p(\Omega; \mathbf{X}))}{\partial \mathbf{X}} \right] = 0 \quad \text{for all } \mathbf{X}. \quad (6)$$

Equation 6 states that the optimal estimator given a set of data is the local maximum of the log-likelihood function. When the condition in Equation 6 is met, the lower bound of the variance in the estimation is

$$var(\hat{\mathbf{X}}) \geq \frac{1}{-E \left[ \frac{\partial^2 \ln(p(\mathbf{\Omega}; \mathbf{X}))}{\partial \mathbf{X}^2} \right]}. \quad (7)$$

The denominator of equation 7 is also known as the Fisher information. Therefore, the CRLB can also be described as the reciprocal of the Fisher information [13]. Using  $I(\mathbf{X})$  to represent the Fisher information matrix,

$$var(\hat{\mathbf{X}}) \geq \frac{1}{I(\mathbf{X})}. \quad (8)$$

Describing the CRLB as the reciprocal of the information matrix, we can make the statement that the more information given by the measurements, the less variance we would expect in the estimation, thus the more confident we are in the calculated estimate. An increase in information for this research can occur with due to an increase in measurements, or a decrease in the variance assumed for a given measurements. Because the variance on a measurement is dependent on the sensor used to create an estimate, we can increase the number of measurements used or use sensors determined to have lower measurement variances.

Knowing the CRLB for the NLO algorithm allows for the ability to determine the amount of variation or known uncertainty for a given estimate. The CRLB for the weighted Gauss-Newton method is known from [14][15] as

$$\Sigma_{\mathbf{X}} = (\mathbf{J}^T \Sigma_{\Omega}^{-1} \mathbf{J})^{-1}. \quad (9)$$

Equation 9 shows that the minimum amount of variance on an estimation using the NLO is the first half of equation 4; therefore, we say that the CRLB is inherently

calculated when estimating an object state. Because the CRLB represents the amount of variance expected on the estimator, it can be used to mathematically represent the amount of uncertainty we have in the state estimate.

Error on a sensor can be represented as the amount of variation on the measurements from the truth. Therefore, error can be represented by a set of variance values. Each measurement has two values, azimuth and elevation. The variances on the measurements are set up in a covariance matrix  $\Sigma_\Omega$  where the values of the covariance matrix are dependent on the sensor itself and are seen as inherent qualities of the sensor. Different sensors are described by different covariance matrices that stay constant over time and are known prior to operational use. The multiple covariance matrices are combined using equation 9 to find the amount of error on an estimate, and the resulting covariance matrix is depicted for analysis on a given state estimate.

The depiction of a covariance matrix that represents the Gaussian distribution is done using ellipsoids [16]. For this practice, the ellipsoid created from the covariance matrix defines a confidence region  $g$  defined by

$$(\mathbf{X} - \hat{\mathbf{X}})^\top \Sigma_X^{-1} (\mathbf{X} - \hat{\mathbf{X}}) = g^2. \quad (10)$$

Equation 10 normalizes the difference between the true state  $\mathbf{X}$  and the estimated state  $\hat{\mathbf{X}}$  using the inverse of the state's covariance matrix  $\Sigma_X$ . To produce a covariance ellipsoid, we calculate multiple points on the ellipsoid such that each point yields the same value for  $g$ . If we desire a 3-sigma ellipsoid, then we want to calculate the  $\mathbf{X}$  values that lie on an ellipsoid that yield a value of three for  $g$ .

To calculate a given confidence region, the covariance matrix  $\Sigma_X$  is effectively decomposed into two equal triangular matrices  $\mathbf{L}$  using the Cholesky decomposition

$$Cholesky(\Sigma_X) = \mathbf{L}\mathbf{L}^\top. \quad (11)$$

Cholesky decomposition is analogous to taking the square root of a scalar value, but for a positive semi definite matrix. Based on equation 10, the covariance ellipsoids are produced by first declaring multiple points on a sphere  $\mathcal{S}$  via

$$\mathcal{S} = Sphere(radius = 1). \quad (12)$$

The group of points on the sphere  $\mathcal{S}$  are then multiplied by the upper triangular matrix  $\mathbf{L}$  and scaled by a standard deviation value  $\sigma$  to contour the points into the desired confidence region or covariance ellipsoid

$$= \sigma \mathbf{S} \mathbf{L}. \quad (13)$$

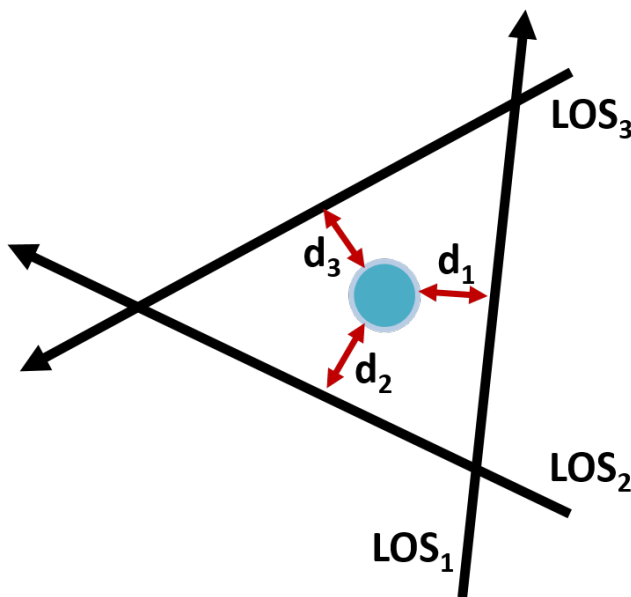
The standard deviation value  $\sigma$  defines how many standard deviations of error are encompassed in the covariance ellipsoid. Using equation 13, the state covariance matrix calculated in equation 9 is visualized. The visualization of the state covariance matrix as an ellipsoid allows us to represent the region that, with a desired probability, we are confident an object resides in. Thus, we can be aware of any possible interactions an object may have with its surroundings.

## 2.4 Geolocation Algorithms

In different scenarios an object of interest will display different characteristics. Therefore, it is necessary to take into account different assumptions about the object. For example, if an object is stationary, then over time the measurements on the object can be assumed to remain constant. When using multiple non-time coincident measurements to create a single estimation. There is no need to translate the measurements in time based on the dynamics of the object. Therefore, different simulations may show different algorithms as more effective.

NLO is an iterative algorithm that requires an initial state estimate to begin. For the NLO to converge onto an estimate, it is imperative to have a good initial estimate to avoid the possibility of divergence on the iterations [15]. One method of producing an initial estimate is using a triangulation algorithm.

The triangulation algorithm is a non-iterative method that produces an estimate based on the least mean squared error between the LOS vectors and an estimated position as shown in Figure 5. The triangulation algorithm works to minimize a



**Figure 5.** Depiction of the triangulation algorithm from line of sight vector measurements

squared distance  $D$  using  $N$  measurements, weights  $w_n$ , and the associated distances between the estimate and the LOS measurements  $d_n$  via

$$D = \sum_{n=1}^N d_n^2 w_n. \quad (14)$$

The triangulation algorithm that minimizes  $D$  in equation 14 is similar to those used in [17] and [18].

Using the initial estimate calculated from the sensor measurements, the next step

is to calculate the change in the the measurements  $\Delta\Omega$ . To calculate the measurements that would result in the initial estimation  $\Omega$  is found from the measured position  $\mathbf{P}_{meas}$  via

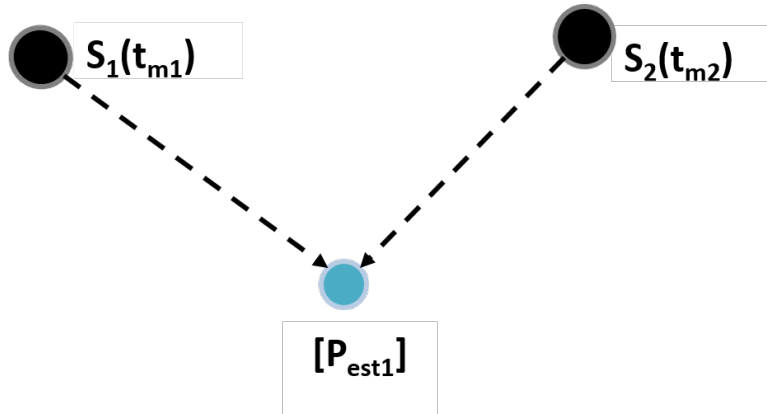
$$\begin{aligned}\Omega(\mathbf{X}) &= [\theta(\mathbf{X}), \phi(\mathbf{X})]^\top, \\ &= \left[ \tan^{-1} \left( \frac{\mathbf{P}_{meas}(y)}{\mathbf{P}_{meas}(x)} \right), \tan^{-1} \left( \frac{\sqrt{\mathbf{P}_{meas}(x)^2 + \mathbf{P}_{meas}(y)^2}}{\mathbf{P}_{meas}(z)} \right) \right]^\top.\end{aligned}\quad (15)$$

Therefore,  $\Delta\Omega$  is the difference between the actual measurements and the measurement vector calculated from the estimated position.

When an object is assumed to be static, the resulting estimated position  $\mathbf{P}_{est}$  is related to the measured position  $\mathbf{P}_{meas}$  by the equality

$$\mathbf{P}_{meas} = \mathbf{P}_{est}.\quad (16)$$

Equation 16 assumes that an object is in the same position for every measurement taken independent of the time it was taken at. The depiction of the assumption is shown in Figure 6.



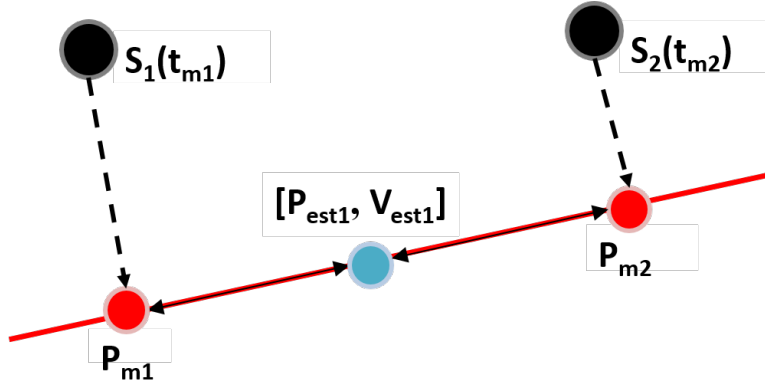
**Figure 6. Visualization of the static non-linear optimization assumptions**

If the localized object is moving at a constant velocity, then the position of the

object is dependent on the time at which the measurement is taken at. Therefore, when multiple measurements are used to produce a single estimate. Using  $\mathbf{V}_{est}$  as the velocity at the estimation time and  $\tau$  as an electromagnetic propagation factor, the measured position and the estimation position are related via

$$\begin{aligned}\mathbf{P}_{meas} &= \mathbf{P}_{est} - \int_{(t_{meas}-\tau)}^{t_{est}} \mathbf{V}_{est} dt, \\ \mathbf{P}_{meas} &= \mathbf{P}_{est} - \mathbf{V}_{est}(t_{est} - (t_{meas} - \tau)).\end{aligned}\tag{17}$$

Equation 17 shows that for  $\mathbf{V}_{est}$ , the position of an object is related to the measurement time  $t_{meas}$ , the estimation time  $t_{est}$ , and a time delay factor  $\tau$  based on the propagation delay of electromagnetic radiation [1]. The relation between the measurement positions and estimated position are shown in Figure 7.



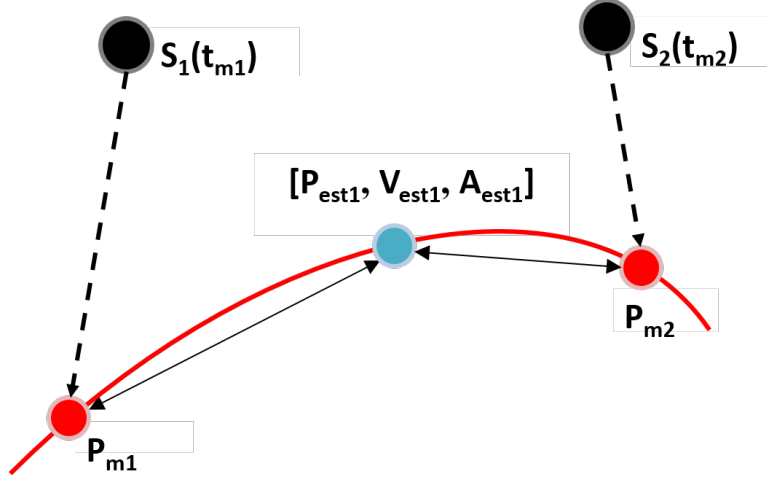
**Figure 7. Visualization of the velocity non-linear optimization assumptions**

To better understand the dynamics of an object, acceleration is included with the position and velocity estimates within the state vector. Objects are influenced by external forces, notably gravity. When forces act on an object in space, there will be a corresponding acceleration associated with it. Therefore, if we no longer assume constant velocity within a window of time, using  $\mathbf{A}_{est}$  to represent the estimated

acceleration, the dynamics of an object are described using the equations

$$\begin{aligned} \mathbf{P}_{meas} &= \mathbf{P}_{est} + \int_{t_{est}}^{(t_{meas}-\tau)} \mathbf{A}_{est}t + \mathbf{V}_{est}dt, \\ \mathbf{P}_{meas} &= \mathbf{P}_{est} + \mathbf{V}_{est}((t_{meas} - \tau) - t_{est}) + \mathbf{A}_{est} \left( \frac{(t_{meas} - \tau)^2 - t_{est}^2}{2} \right). \end{aligned} \quad (18)$$

Equation 18 assumes that over a window of time defining the measurements used to create an estimation that the acceleration is approximately constant. Figure 8 depicts the new path assumption of the object. It may not always be beneficial to increase



**Figure 8. Visualization of acceleration and velocity non-linear optimization assumptions**

the complexity of calculations with the Acceleration and Velocity NLO if the window of measurements used to find an estimated position is small. The decrease in benefits from the algorithm is due to the lack of acceleration experienced within a window of time. The smaller the window, the smaller the change in position for an object can be assumed because the difference in the estimation time and measured times makes the acceleration and velocity insignificant. Therefore, it is necessary to take into account the number of measurements over time and the dynamics of the object to determine the most suitable algorithm for localization.

## 2.5 Bias

Bias with respect to the research is a time correlated random process adding an offset to a measurement. A bias is defined to be a constant offset on a given estimation [13]. In this research, the term bias is used to define the random process because within a small window of time, the error provides a relatively constant offset. Different windows therefore have different biases dictated by our process. Because we handle bias as a random process, the bias may change over time and is not necessarily stationary. The typical way of depicting the error on a sensor for a localization problem is to combine the bias error with a random process model by an IID zero mean Gaussian distribution [19] referred to within the research as noise. For a remote sensor, many possible sources contribute to a bias within a system.

One source of bias error on a remote sensor is the IMU. An IMU primarily consists of a gyroscope and an accelerometer. These components have multiple types of errors such as misalignments and scale factors, but more importantly bias errors [20][21]. There have been multiple attempts within either software or hardware to mitigate the errors from these components by modeling the error with Markov models and estimating the bias using a Kalman filter [22][23][24]. One limitation of modeling the bias within an algorithm is that it does not account for calibrations of the bias that might occur over the period an object is being localized. Calibrations are used to correct the biasing errors that accumulate over time. The corrections may occur at unknown times adding complexity to the bias estimation problem. A common calibration method for satellite based remote sensors is to use star trackers [25] and other processing means. In addition, the IMU measurements are integrated to estimate the quantification used in this research making the output curve of the IMU time correlated.

An accelerometer on an IMU is used to measure accelerations occurring on a

remote sensor. The acceleration measurements are converted to positional values via a double integration where  $\epsilon_{Bias}$  represents bias error at the measurement times and  $A_{error}$  references error in the acceleration measurement depicted mathematically as

$$\epsilon_{Bias} = \int_0^{t_{meas}} \int_0^{t_{meas}} A_{error} dt dt. \quad (19)$$

Because a sensor system takes discrete samples from the accelerometer to produce positional estimates, the positional bias in the measurements can be discretely defined by the double summation of the number of samples  $N$ , via

$$\epsilon_{Bias} = \sum_{n=1}^N \sum_{m=1}^n A_{error,m}. \quad (20)$$

$A_{error}$  follows a random distribution that accumulates over time. Every time a sample of the accelerometer is taken, the analog signal passes through an analog to digital converter (ATD) converter. Error is produced in the truncation of the analog signal to a discrete value when quantizing in the ATD[26]. Figure 9 shows the effects of quantizing on an analog signal.

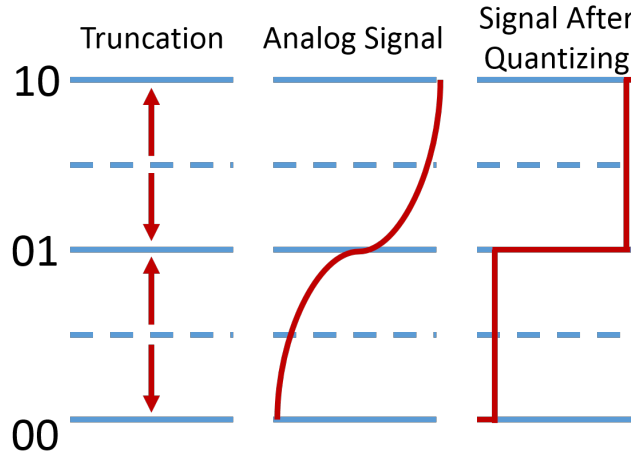
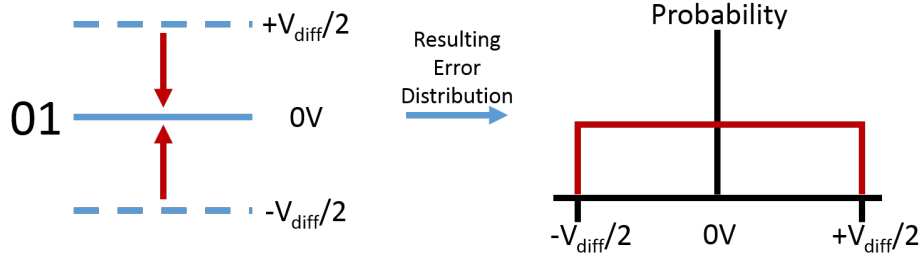


Figure 9. Truncation of signal due to analog-to-digital quantization

As described in Figure 9, the ATD changes the analog signal thus inducing error equal to the amount of truncation within the measurements. A signal is assumed to follow a uniform random distribution within the dynamic range of the ATD. Therefore, if every quantization level has a voltage range  $V_{diff}$  that is assigned to it, and the error associated to every sample can be described by the distribution in Figure 10.



**Figure 10. Probabilistic effects on error due to analog-to-digital quantization**

From Figure 10, the induced error from the ATD converter can be represented by a uniform distribution that ranges across a voltage difference. The error can be mathematically described via

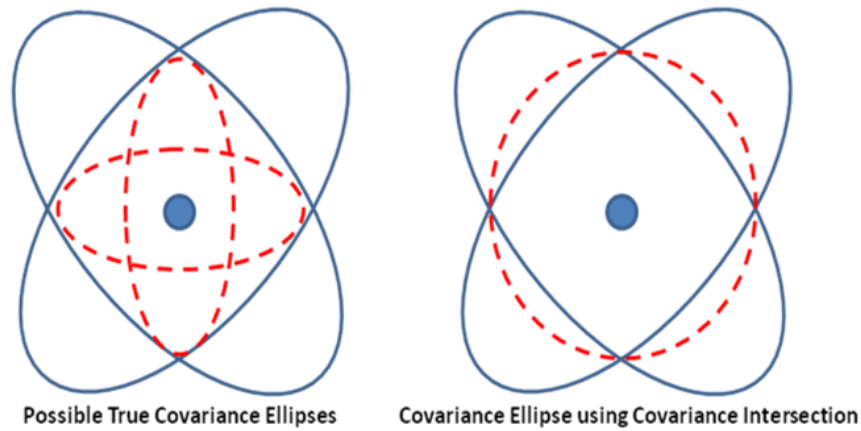
$$A_{error} = \mathcal{U}(-V_{diff}/2, V_{diff}/2). \quad (21)$$

Equation 21 when paired with equation 20 determines the simulated bias for use with the localization algorithms. Remote sensors are known to have some amount of bias in the measurements due to the drifting IMU measurements. Because bias is known, the sensors often calibrate themselves while in flight [27][28].

## 2.6 Covariance Intersection

One method of accounting for a bias in multiple measurements is assuming the measurements are correlated with one another. When error is perfectly correlated

between two measurements, the error on one measurement is the same as the error on the next measurement. An algorithm known as CI is used to take this assumption into account within the NLO. CI is intended for use when the amount of correlation between measurements is unknown [29][30]. The CI algorithm introduced in 1997 uses a generalized method of combining measurements without the loss of information. CI works under the idea that if the correlation between two measurements is unknown then no matter what the cross-correlation terms are, the true covariance ellipsoids that results from the combination of the two measurements lies somewhere within the intersection of the individual covariance matrices for the measurements. When describing the covariance matrices geometrically as covariance ellipsoids, when the cross-correlation terms are unknown, for the calculated covariance ellipsoids to consistently encompass the true covariance ellipsoids, the calculated covariance ellipsoids needs to encompass the intersection of the individual covariance ellipsoids. Figure 11 depicts how CI combines two individual covariance matrices represented by solid blue ellipsoids into a single covariance ellipse represented by the dashed red ellipsoids that encompasses all possible true ellipsoids.

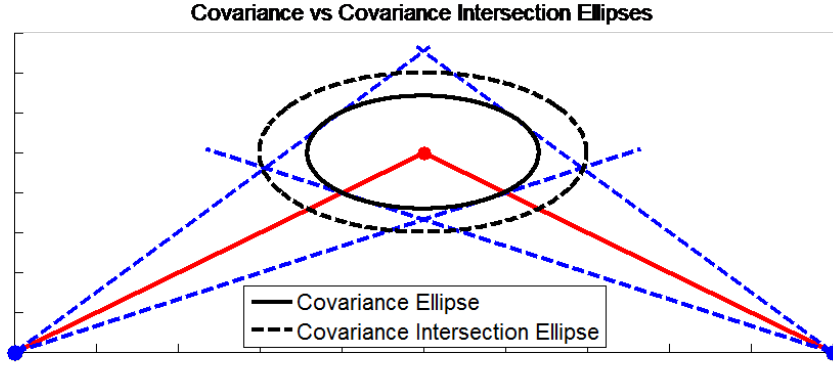


**Figure 11.** Possible covariance ellipsoids (left) and the corresponding covariance ellipse using CI (right)

Because all possible true covariance ellipsoids are within the intersection of the

individual measurements covariance ellipsoids, a conservative estimate for the true error is an ellipsoid that encompasses the entire intersection.

The geometric representation of a single measurement's covariance ellipsoid cannot truly be realized. A single measurement has two pieces of information, the azimuth and elevation angle projecting from the sensor's position, while an ellipse exists within a three dimensional space. Therefore, a measurement's error is depicted as a range diverging from the true LOS due to a difference in angle as shown in Figure 12.



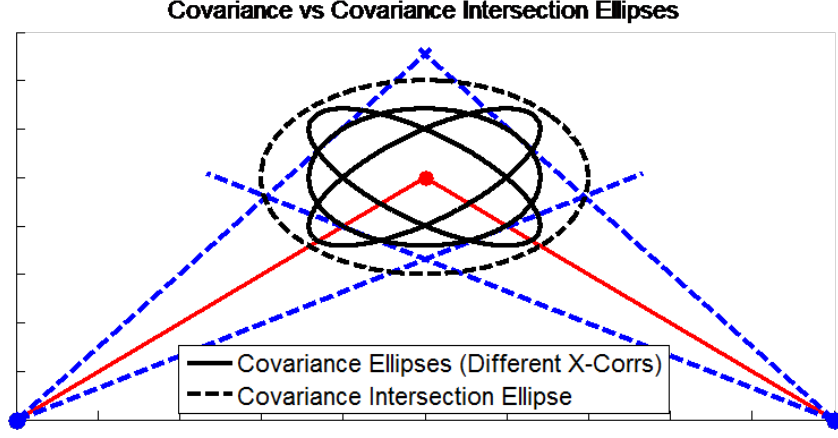
**Figure 12.** Resulting covariance ellipsoids (black) from two angle of arrival measurements with depicted error (blue) on an angle of arrival measurement (red)

When using different cross correlation values in the covariance matrices, the Covariance Intersection algorithm is still able to engulf all possibilities as shown in Figure 13.

The intersection of the two measurements is the convex combination of the covariance matrices. Using  $\Sigma_c$  as the calculated covariance matrix,  $\Sigma_1$  to  $\Sigma_N$  as the individual covariance matrices,  $\bar{\mathbf{m}}_c$  as the calculated mean,  $\bar{\mathbf{m}}_1$  to  $\bar{\mathbf{m}}_N$  as the individual means, and  $\omega_1$  to  $\omega_N$  as the scalar weighting factors, CI is described as

$$\Sigma_c^{-1} = \sum_{n=1}^N \omega_n \Sigma_n^{-1}, \quad (22)$$

$$\Sigma_c^{-1} \bar{\mathbf{m}}_c = \sum_{n=1}^N \omega_n \Sigma_n^{-1} \bar{\mathbf{m}}_n. \quad (23)$$



**Figure 13.** Resulting covariance ellipsoids (black) from two angle of arrival measurements with depicted error (blue) on an angle of arrival measurement (red) when true covariance ellipse has different amounts of correlations

Equation 23 shows that the mean value  $\overline{\mathbf{m}}_c$  also changes depending on the amount of correlation assumed within the measurements.

## 2.7 Path Optimization

Localization can refer to finding a single point estimation, but may also refer to the calculation of multiple positions over a length of time. The multiple positions as a group are referred to as the object's path. Within the NLO, the desire is to append a position to position transition matrix to the Jacobian matrix. The purpose of the transition matrix is to optimize the continuity of the position and velocity estimates in conjunction with the AoA measurements. For a given state estimate with a position and velocity, the position of the next estimated state should be the sum of the position and the distance traveled between the current and next states based on the estimated velocity. Figure 14 shows position estimates from the NLO in blue and estimated positions from the previous state in red.

From Figure 14, the position calculated from the values of the previous state may not agree with the state estimate from the AoA measurements. The transition from

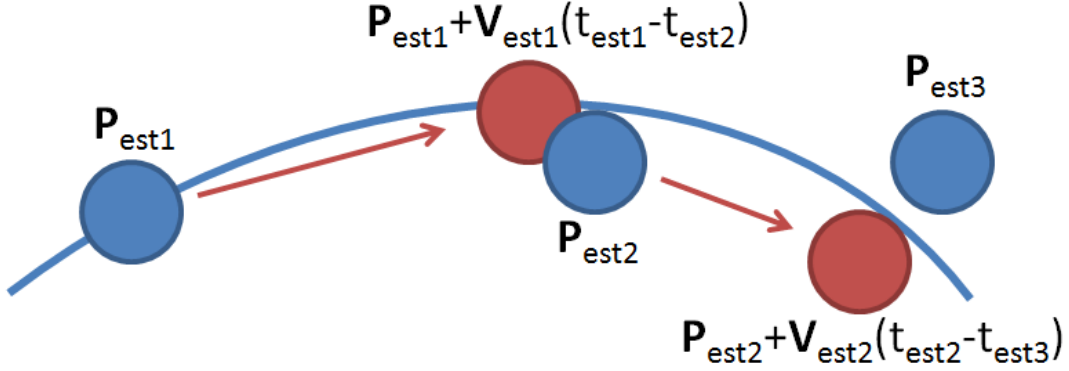


Figure 14. Depiction of the transition matrix (red) compared to position estimates based solely on measurements (blue)

one state to the next is expressed as

$$\mathbf{P}_{est1} + \mathbf{V}_{est1} * (t_{est2} - t_{est1}) \neq \mathbf{P}_{est2}. \quad (24)$$

Because an object following a given path will need to follow a kinematic model, the NLO can acquire more accurate position estimation by optimizing the equation

$$0 \approx \mathbf{P}_{est1} + \mathbf{V}_{est1} * (t_{est2} - t_{est1}) - \mathbf{P}_{est2}. \quad (25)$$

Equation 25 is the basis for the transition model being appended to the Jacobian in the NLO. The transition model smooths anomalous estimates from imprecise measurements to the rest of the estimated data points creating a continuous track. The assumption is for a given time period, the object being localized has a continuous velocity and acceleration.

## 2.8 Analysis

In order to test the localization algorithms, a number of simulations are randomly created. The use of multiple random simulations in order to benchmark an algo-

rithm is called a Monte-Carlo simulation (MCS). MCS are used to verify whether the algorithms perform under a multitude of scenarios. Different parameters are held constant as others fluctuate to analyze the effects the of different variables on an algorithm. Several metrics of the MCS outputs can be used to measure the effectiveness of the NLO algorithm. These include average normalized estimation error squared (ANEES) [16], MSE, and average covariance ellipsoid size. Each of these are described in more detail below.

The ANEES defines how well the estimated position and estimated state covariance matrix agree with one another for the given set of MCS. The first part of calculating the ANEES is finding the difference between the true object location  $\mathbf{X}$  and the estimated location  $\hat{\mathbf{X}}$  as

$$\tilde{\mathbf{X}} = \mathbf{X} - \hat{\mathbf{X}}. \quad (26)$$

The difference in the true and estimated state is normalized to the estimated state covariance matrix

$$\epsilon_{nees} = \tilde{\mathbf{X}}^\top \Sigma_X^{-1} \tilde{\mathbf{X}}. \quad (27)$$

The ANEES is therefore the expected value of the NEES over a large amount of random simulations, the MCS. The ANEES is important because if the number of degrees of freedom  $n_X$  is equal to the ANEES, the estimator is said to be consistent. Mathematically this equality is described as

$$E[\epsilon_{nees}] = n_X. \quad (28)$$

The ANEES is able to tell us if the estimated covariance matrix properly represents the estimate that is calculated. The ANEES does not take into account the ability of the estimator to produce the most accurate estimates, or decrease the sizes of the

state covariance matrices as feasibility allows.

The next means of analyzing the localization algorithms is by calculating the MSE of the estimates. Calculating the MSE for the MCS allows for the ability to compare the average MSE. The MSE is calculated via

$$MSE = (\mathbf{X} - \hat{\mathbf{X}})^\top (\mathbf{X} - \hat{\mathbf{X}}). \quad (29)$$

The average MSE for the MCS is the expected value for all of the calculated MSEs. The MSE is used to determine which algorithms produce the most accurate results independent of the covariance matrix.

The final analysis of the algorithms is the calculation of the average covariance ellipsoid size for each estimate. There are multiple ways to define the size of an ellipsoid. For this application, we will look at the semi-major and semi-minor axes of the ellipsoids. The semi-major and semi-minor axes depict the shortest and farthest distances away from the estimate to the edge of the ellipsoids. If an ellipsoid is stretched, this may not be apparent by calculating the volume of the ellipse; therefore, it is beneficial to analyze the ellipsoid with the calculated axes. The semi-major and semi-minor axes are calculated using a singular value decomposition (SVD)[4]. Assuming a  $3 \times 3$  covariance matrix  $\mathbf{\Sigma}_X$  its SVD is

$$\mathbf{\Sigma}_X = \mathbf{U}\mathbf{S}\mathbf{B}^*,$$

$$\mathbf{S} = \begin{bmatrix} \text{Semi-Major Axis} & 0 & 0 \\ 0 & \text{3rd Normal Axis} & 0 \\ 0 & 0 & \text{Semi-Minor Axis} \end{bmatrix}. \quad (30)$$

The matrices  $\mathbf{U}$  and  $\mathbf{B}$  are undesired matrices for the analysis from the decomposition,

but represent the rotation of the ellipsoids.

## 2.9 Conclusion

The theoretical overview of the research focuses on localization along with the error calculations for the remote sensors being used. The purpose for focusing on NLO is the inherent ability for the calculation to produce the state estimation covariance matrix from equation 13. To produce more accurate error estimations, the focus of the research will use a transition matrix to localize an object along its entire path. The assumption is that an object follows a kinematic model for its motion. The Covariance Intersection method will be investigated in order to create error ellipsoids that encompass the true error parameter on the position estimation. These methods will be tested to see if they provide better results for an object's localization compared to previous localization techniques.

### III. Methodology

#### 3.1 Introduction

The localization algorithm employs remote sensor AoA information. Previous iterations of the algorithm relied on uncorrelated measurements from multiple sensors. The focus of this research is to look at the case where sensor measurements have correlated error. The research investigates how to deal with correlated error in an attempt to quantify it within a covariance ellipsoid depicting confidence on a state estimation.

Previous research in the localization algorithm assumed the error on the measurements was due to random noise alone [1]. The problem with this assumption was that measurements from a single sensor are often correlated. Error correlation is due to sensor bias. A bias causes measurements from a sensor to have similar error; therefore, the expected value of error is not zero, and the measurements are correlated. If the bias becomes the dominant source of error, the localization algorithm will produce state estimates that are constantly shifted off the true state.

The investigation explores the data simulation with the data composed of true AoA along with the apparent AoA that has the addition of error from random noise and a bias. The AoA is generated from a simulated object path and simulated sensor positions. The different variations of the localization algorithm are tested against multiple simulated object paths and error quantities to compare the results with the older variations of the algorithm. The intent of the evaluation is to determine how the older variations of the NLO deteriorate in performance with the new simulations along investigating how well the updates to the algorithm are able to mitigate these effects.

### 3.2 Theory

As discussed in Chapter II, the base algorithm for localization is called NLO. In equation 4, the NLO is calculated using the Jacobian matrix of the measured AoA values with respect to the state variables for a specified estimation time. The covariance matrix for the sensors used in the NLO is assumed to describe a Gaussian distribution. No other distribution can be fully described using only a variance parameter. The assumption is no longer valid when there is correlation in a sensor's measurements.

#### Acceleration Estimation.

Three variants of the NLO exist in terms of the state vector being estimated. The Static NLO estimates the Cartesian coordinates of an object, and the Velocity NLO estimates the position vector along with the velocity vector, and the Acceleration and Velocity NLO estimates the position, velocity, and acceleration vectors in Cartesian coordinates space as shown in equation 31 where  $x$ ,  $y$ , and  $z$  are the variables describing position,  $\dot{x}$ ,  $\dot{y}$ , and  $\dot{z}$  are the variables describing the velocity, and  $\ddot{x}$ ,  $\ddot{y}$ , and  $\ddot{z}$  are the variables describing acceleration in the corresponding directions.

$$\mathbf{X}_{stat} = [x, y, z]^\top, \quad \mathbf{X}_{vnlo} = [x, y, z, \dot{x}, \dot{y}, \dot{z}]^\top, \quad \mathbf{X}_{avnlo} = [x, y, z, \dot{x}, \dot{y}, \dot{z}, \ddot{x}, \ddot{y}, \ddot{z}]^\top. \quad (31)$$

The state estimates are calculated over a window of time. The window is a subsection of the overall object path; therefore, each state estimate uses a portion of the overall measurements.

The calculations for the NLO require an initial estimate from the triangulation algorithm. The triangulation algorithm estimates the position and not the velocity or acceleration. A quick method of producing estimates for the velocity and acceleration

is to calculate the change in position with respect to time  $t$  as

$$\begin{aligned}\dot{x}_{est1} &= \frac{x_{est1} - x_{est2}}{t_{est1} - t_{est2}}, \\ \ddot{x}_{est1} &= \frac{\dot{x}_{est1} - \dot{x}_{est2}}{t_{est1} - t_{est2}}.\end{aligned}\tag{32}$$

The initial estimates therefore require at least the triangulation position estimates created from different subsets of measurements.

The NLO requires the calculation of the Jacobian matrix. Each cell in the matrix is the partial derivative of one of the measurement terms with respect to one of the values in the estimated state vector. When calculating multiple derivatives with respect to the acceleration values, it is more computationally efficient to estimate the derivatives numerically. Because the derivatives are estimated, some amount of error will be assumed within the calculations. The equation for the derivative of the function  $f(x)$  is defined as

$$\frac{\partial f(x)}{\partial x} = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}.\tag{33}$$

When estimating the derivative of a function, the  $h$  term in equation 33 is set to some small number instead of evaluating the limit. The estimate incurs an error on the order of  $h$  via

$$hf''(x) + O(h^2) \text{ [31]}.\tag{34}$$

This method of calculating derivatives creates relatively little error depending on the value chosen for  $h$ . However, calculating the same number of points from the function  $f(x)$  a more accurate estimate of the derivative can be calculated.

Instead of calculating the derivative using  $f(x)$  and a value a set distance away

$f(x + h)$ , a better derivative estimate can be created using two points near  $f(x)$  as

$$\frac{\partial f(x)}{\partial x} \approx \frac{f(x + h) - f(x - h)}{2h}. \quad (35)$$

The derivative estimate in this calculation has an error calculated by the equation

$$\frac{1}{6}h^2 f'''(x) + O(h^3) \text{ [31]}.$$

Equation 35 to calculate the derivative, the error is on the order of  $h^2$ . When  $h$  is less than one, the error in the estimation is lower than the error from the previous method of calculating the derivative.

We perform the derivative estimate on equation 15 using the calculation of  $P_{meas}$  from equation 18. All partial derivatives for the Jacobian in the acceleration and velocity NLO are populated using this method, simplifying the calculations for the algorithm.

### **Covariance Intersection for Non-Linear Optimization.**

CI is defined by the equations 22 and 23. The algorithm is set up in terms of summations rather than the linear algebra format of the NLO in equation 3. CI is used assuming that the correlation on the measurements is unknown. When the measurement correlation is completely unknown, the CI algorithm can assume perfect correlation between measurements to create a conservative approximation of the covariance ellipse.

First, we want to be able to translate the matrix multiplications into individual summations that calculate individual matrix elements. An element in a matrix  $\mathbf{C}$  will be defined by its row  $m$  and column  $n$  where  $\mathbf{C}$  is equal to the matrix multiplication

of matrices  $\mathbf{A}$  and  $\mathbf{B}$ . Each cell is calculated by the equation

$$c_{mn} = \sum_{i=1}^N a_{mi} b_{in}. \quad (36)$$

Let a new matrix  $\mathbf{D}$  be defined by the equation

$$\mathbf{D} = \mathbf{B}^\top \mathbf{A} \mathbf{B} = \mathbf{B}^\top \mathbf{C}. \quad (37)$$

The transpose is used to reflect a matrix along the diagonal. Within the summation, the reflection can be seen as a reference swap of the row and column indices. The matrix elements of  $\mathbf{D}$  can be written as

$$d_{mn} = \sum_{i=1}^M b_{im} \sum_{j=1}^N a_{ij} b_{jn}. \quad (38)$$

When the matrix  $\mathbf{A}$  is known to be a diagonal matrix where all off-diagonal elements are known to be zero, it can be said that

$$\begin{aligned} & \text{when } m \neq i, a_{mi} = 0, \\ & c_{mn} = \sum_{i=1}^N a_{mi} b_{ik} = a_{mm} b_{mn}, \\ & \therefore d_{mn} = \sum_{i=1}^N b_{im} a_{ii} b_{in}. \end{aligned} \quad (39)$$

In the previous example,  $\mathbf{B}$  is used in place of the Jacobian matrix  $\mathbf{J}$ , and  $\mathbf{A}$  is in place of the covariance matrix  $\mathbf{\Sigma}_\Omega$ . For the NLO, equation 39 can be used to depict the calculation of the localization error  $\mathbf{\Sigma}_X^{-1}$ . Substituting in the proper variables, the

equation becomes

$$\begin{aligned}\Sigma_X^{-1} &= \mathbf{J}^\top \Sigma_\Omega^{-1} \mathbf{J}, \\ \Sigma_X^{-1}(m, n) &= \sum_{i=1}^N \mathbf{J}(i, m) \Sigma_\Omega^{-1}(i, i) \mathbf{J}(i, n).\end{aligned}\tag{40}$$

For equation 40, the elements referenced within each matrix are defined by the indices within the parenthesis next to its corresponding variable.

To isolate the individual sensor measurements in equation 40, the equation can be written as

$$\begin{aligned}\Sigma_X^{-1}(m, n) &= \sum_{i=1}^2 \mathbf{J}(i, m) \Sigma_\Omega^{-1}(i, i) \mathbf{J}(i, n) + \sum_{i=3}^4 \mathbf{J}(i, m) \Sigma_\Omega^{-1}(i, i) \mathbf{J}(i, n), \\ \therefore \Sigma_X^{-1} &= \Sigma_{X, meas1}^{-1} + \Sigma_{X, meas2}^{-1}\end{aligned}\tag{41}$$

Applying weights to the individual covariance matrices on the right-hand side of equation 41, the equation becomes identical to the first part of the definition of the CI algorithm in equation 22 seen as

$$\Sigma_X^{-1} = \omega_1 \Sigma_{X, meas1}^{-1} + \omega_2 \Sigma_{X, meas2}^{-1}.\tag{42}$$

In order to simplify the equation for use within MATLAB, equation 42 can be described as

$$\begin{aligned}\Sigma_X^{-1}(m, n) &= \omega_1 \sum_{i=1}^2 \mathbf{J}(i, m) \Sigma_\Omega^{-1}(i, i) \mathbf{J}(i, n) + \omega_2 \sum_{i=3}^4 \mathbf{J}(i, m) \Sigma_\Omega^{-1}(i, i) \mathbf{J}(i, n), \\ \Sigma_X^{-1}(m, n) &= \sum_{i=1}^2 \mathbf{J}(i, m) \omega_1 \Sigma_\Omega^{-1}(i, i) \mathbf{J}(i, n) + \sum_{i=3}^4 \mathbf{J}(i, m) \omega_2 \Sigma_\Omega^{-1}(i, i) \mathbf{J}(i, n).\end{aligned}\tag{43}$$

A matrix  $\mathbf{W}$  containing the individual weights of the two measurements will be defined

as

$$\mathbf{W} = \begin{bmatrix} \omega_1 & 0 & 0 & 0 \\ 0 & \omega_1 & 0 & 0 \\ 0 & 0 & \omega_2 & 0 \\ 0 & 0 & 0 & \omega_2 \end{bmatrix}. \quad (44)$$

Using this definition of  $\mathbf{W}$ , equation 43 reduces to

$$\begin{aligned} \Sigma_X^{-1}(m, n) &= \sum_{i=1}^N \mathbf{J}(i, m) \mathbf{W}(i, i) \Sigma_\Omega^{-1}(i, i) \mathbf{J}(i, n), \\ \therefore \Sigma_X^{-1} &= \mathbf{J}^\top (\mathbf{W} \Sigma_\Omega^{-1}) \mathbf{J}. \end{aligned} \quad (45)$$

Equation 45 provides a simple method to calculate the covariance matrix depicting the localization error using the CI algorithm. Under the circumstance where errors on the sensor measurements are assumed to be perfectly correlated, the weights are defined as

$$\sum_{i=1}^N \omega_i = 1. \quad (46)$$

CI also requires the estimation of the mean under the same assumptions as the calculation of the covariance matrix depicting the localization error. The equation for  $\Delta x$  can be written as

$$\Delta x = \Sigma_X (\mathbf{J} \Sigma_\Omega^{-1} \Delta \Omega). \quad (47)$$

The value for  $\Sigma_X$  is the inverse of the matrix calculated in equation 45. Converting equation 47 into summation form, the calculation of the individual elements in  $\Delta \mathbf{X}$

becomes

$$\begin{aligned}\Delta \mathbf{X}(m, n) &= \sum_{i=1}^N \Sigma_X(m, i) \left( \sum_{j=1}^M \left( \sum_{k=0}^K \mathbf{J}(k, m) \Sigma_{\Omega}^{-1}(k, j) \right) \Delta \Omega(j, n) \right), \\ &\text{when } k \neq j, \Sigma_{\Omega}^{-1}(k, j) = 0, \\ \therefore \Delta \mathbf{X}(m, n) &= \sum_{i=1}^N \Sigma_X(m, i) \left( \sum_{j=1}^M \mathbf{J}(j, m) \Sigma_{\Omega}^{-1}(j, j) \Delta \Omega(j, n) \right).\end{aligned}\quad (48)$$

$M$  in equation 48 represents the number of rows in the matrices  $\mathbf{J}$  and  $\Delta \Omega$ . Every two rows in these matrices correspond to a single sensor measurement because each measurement contains an azimuth and elevation. Therefore, considering again with the example of two sensor measurements, equation 48 can be written as

$$\begin{aligned}\Delta \mathbf{X}(m, n) &= \sum_{i=1}^N \Sigma_X(m, i) \left( \sum_{j=1}^2 \mathbf{J}(j, m) \Sigma_{\Omega}^{-1}(j, j) \Delta \Omega(j, n) + \sum_{j=3}^4 \mathbf{J}(j, m) \Sigma_{\Omega}^{-1}(j, j) \Delta \Omega(j, n) \right) \\ \Delta \mathbf{X} &= \Sigma_X \left( \mathbf{J}^{\top} \Sigma_{\Omega, meas1}^{-1} \Delta \Omega_{meas1} + \mathbf{J}^{\top} \Sigma_{\Omega, meas2}^{-1} \Delta \Omega_{meas2} \right).\end{aligned}\quad (49)$$

Applying weights to equation 49 gives

$$\Delta x = \Sigma_X \left( \omega_1 \mathbf{J}^{\top} \Sigma_{\Omega, meas1}^{-1} \Delta \Omega_{meas1} + \omega_2 \mathbf{J}^{\top} \Sigma_{\Omega, meas2}^{-1} \Delta \Omega_{meas2} \right). \quad (50)$$

After constructing equation 50, we have an equation that is nearly identical to the definition of CI in equation 22. To obtain the equation in a format that can be applied

to data, we reduce the equation by

$$\begin{aligned}
\Delta \mathbf{X}(m, n) &= \sum_{i=1}^N \Sigma_X(m, i) \left( \omega_1 \sum_{j=1}^2 \mathbf{J}(j, m) \Sigma_{\Omega}^{-1}(j, j) \Delta \Omega(j, n) + \omega_2 \sum_{j=3}^4 \mathbf{J}(j, m) \Sigma_{\Omega}^{-1}(j, j) \Delta \Omega(j, n) \right), \\
\Delta \mathbf{X}(m, n) &= \sum_{i=1}^N \Sigma_X(m, i) \left( \sum_{j=1}^2 \mathbf{J}(j, m) \omega_1 \Sigma_{\Omega}^{-1}(j, j) \Delta \Omega(j, n) + \sum_{j=3}^4 \mathbf{J}(j, m) \omega_2 \Sigma_{\Omega}^{-1}(j, j) \Delta \Omega(j, n) \right), \\
\Delta \mathbf{X}(m, n) &= \sum_{i=1}^N \Sigma_X(m, i) \left( \sum_{j=1}^4 \mathbf{J}(j, m) \mathbf{W}(j, j) \Sigma_{\Omega}^{-1}(j, j) \Delta \Omega(j, n) \right), \\
\Delta \mathbf{X} &= \Sigma_X(\mathbf{J}(\mathbf{W} \Sigma_{\Omega}^{-1}) \Delta \Omega), \\
\therefore \Delta \mathbf{X} &= (\mathbf{J}^{\top} (\mathbf{W} \Sigma_{\Omega}^{-1}) \mathbf{J})^{-1} (\mathbf{J}(\mathbf{W} \Sigma_{\Omega}^{-1}) \Delta \Omega). \tag{51}
\end{aligned}$$

Equation 51 is the NLO algorithm using CI. The algorithm is used to estimate an object's state vector under unknown correlation of sensor measurements.

### Transition Matrices.

The transition matrix is used to draw relationships between the multiple estimated object states. Therefore, the transition matrix on each iteration is used to optimize object positions over an entire path. Previously, the NLO algorithms optimized the state of an object one position at a time. The idea behind the transition matrix is that we are estimating the same object over a given period of time. Therefore, we can assume that the state estimate at one position in time can be related to the state estimate at the next position in time.

Previous NLO algorithms would produce multiple state estimates. When examining the overall path created for an object, the covariance ellipsoids would range in size from state to state. If within a window of measurements we were unable to get measurements from multiple different sensors or if all of the measurements came from a similar location, large skewed ellipsoids would be produced. By relating an estimated

state with similar states, each state estimate is incorporating information from both the measurements used to produce its estimates, but also the measurements used to produce other estimates. If a bias is assumed to be near zero over an entire path, even though it may not be zero over a small window of time, the transition matrix will lessen the effects of bias from estimate to estimate.

The transition matrix in one dimension  $x$  will follow the form:

$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}\Delta t^2 & -1 & 0 & 0 \\ 0 & 1 & \Delta t & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ \ddot{x}_1 \\ x_2 \\ \dot{x}_2 \\ \ddot{x}_2 \end{bmatrix}. \quad (52)$$

In this matrix, the value of  $x$  is being calculated for the next estimation time given the change in time  $\Delta t$  and a kinematic model (KM). As described in equation 25, we relate state one with state two by calculating the position, velocity, and acceleration expected for state two and subtracting out what was actually calculated for state two. Doing so, we update the state estimates based on this transition matrix. The expected value that the equation will be optimized to is zero for all cases. The values in this model, in order to work with the rest of the NLO, need associative variances to weight the confidences in the KM. These weights in the simulations will be based on the maximum expected values for an object's velocity and acceleration. Therefore, any state estimates that do not coincide within the realm of a realistic object will be smoothed to fit the model. The amount of smoothing from one point to another is dependent on the amount of error in a state estimate.

The transition matrix representing a kinematic model is concatenated to the Jaco-

bian matrix within the NLO algorithm. Covariance values representing the expected error in the model are also concatenated to the sensor measurement's covariance matrix. Within the NLO algorithm this can be described using  $\mathbf{J}_{trans}$  and  $\Sigma_{trans}$  to describe the transition matrix and transition covariance matrix respectively via

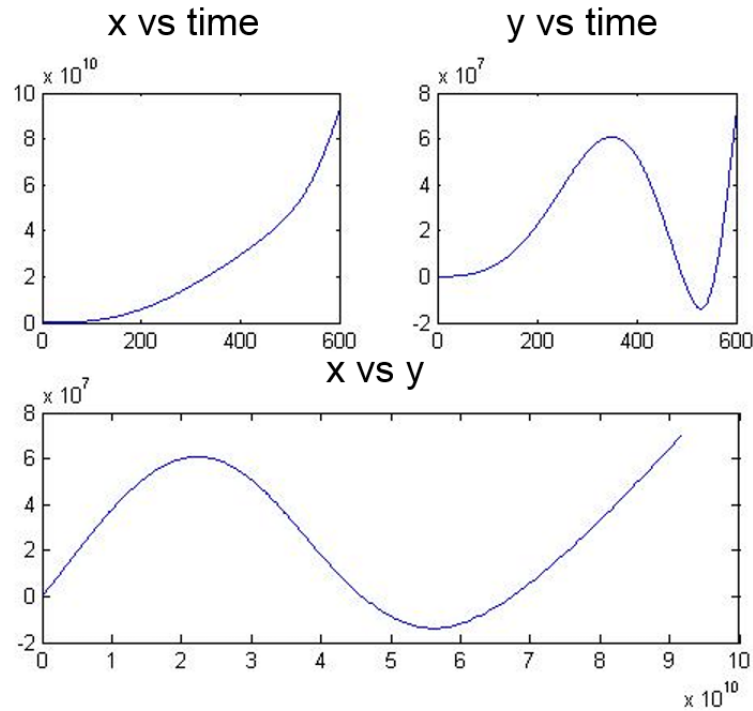
$$\Delta \mathbf{X} = \left( \begin{bmatrix} \mathbf{J} \\ \mathbf{J}_{trans} \end{bmatrix}^\top \begin{bmatrix} \Sigma_\Omega \\ \Sigma_{trans} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{J} \\ \mathbf{J}_{trans} \end{bmatrix} \right)^{-1} \begin{bmatrix} \mathbf{J} \\ \mathbf{J}_{trans} \end{bmatrix}^\top \begin{bmatrix} \Sigma_\Omega \\ \Sigma_{trans} \end{bmatrix}^{-1} \Delta \Omega.$$

The transition covariance matrix uses values that represent the maximum acceleration and velocity an object may exhibit. Because the values are large,  $\Sigma_{trans}$  describes the possible error for the KM as large. Therefore, the KM is not being used to fit the position estimates to a given path that may describe only a specific set of objects. However, the NLO algorithm is still dependent primarily on the measurements from the sensors. One problem that may arise from using large covariance values is the extra complexity added to the NLO algorithm.

### 3.3 Materials and Equipment

The simulations for this research are conducted primarily in the MATLAB environment. MATLAB enables us to create mathematical models for object paths, and create measurements from sensors along the path. Sensors are modeled in MATLAB with elliptical orbits around a fixed earth. The velocities for the sensors are appropriated based on their elevation and the necessary velocity at the elevation to correctly model the orbit. Sensors with a lower elevation therefore travel at faster angular velocities. The object of interest is modeled in two different ways. The first model is of a low earth orbit (LEO) object with the same equations as the modeled sensors. This object model provides a basic path to localize with the NLO. Velocity is constant, and the NLO is expected to produce more accurate results compared to

an object with a more complex path. The second object model is meant to test the limits of the NLO algorithm by having highly erratic motion. The erratic motion is modeled using differential equations described by the forces that act on the object in a real scenario. Solving for the differential equations required the ordinary differential equations (ODE) solver in MATLAB. Using the differential equations, we are able to simulate different amounts of force over time creating a path similar to the one shown in Figure 15.



**Figure 15. Object Path simulated using differential equations**

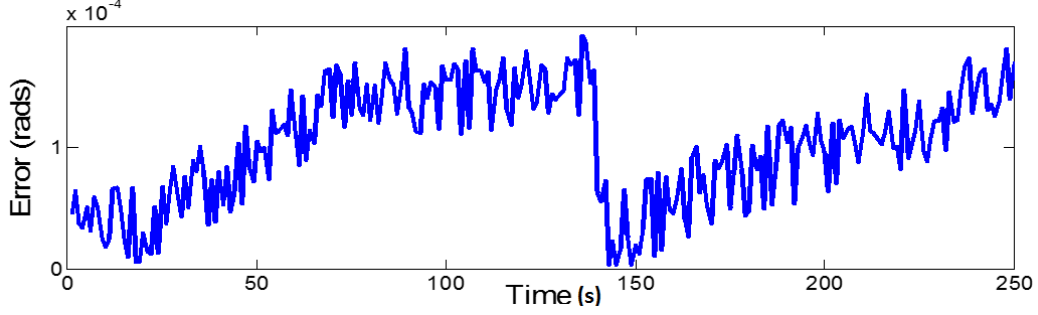
Once the paths of the object and the sensors are created, the measurements are simulated at different times along the path. For the simulations, the beginning and end times are found for all of the paths. Visibility from the sensor to the object is taken into consideration because the earth may eclipse the LOS from the sensor to the object. Next, random measurement times following a uniform distribution are created and assigned to the multiple sensors. Different points in the path are created

at random measurement times. The random measurement times simulate gaps in time where the object of interest may have only a few sensor measurements. The gaps in time create estimates with greater amounts of error during the time period. At the given measurement times, LOS vectors are created between the sensors and the object. Each calculated vector is then assigned to the corresponding sensors, creating an AoA measurement array. The measurements at this point in the simulation do not have any error associated with them.

Sensor error in MATLAB is created in two parts. The first part of the error contains the uncorrelated term, referred to as random noise. The random noise is modeled using IID Gaussian noise. The noise with a known variance is then added to the AoA components in the measurement. The second part of error is attributed to the correlated term modeled with an accumulating bias. Modeled error is assumed to be associated to the truncations of a signal in the accelerometer on the sensor used to create the LOS vector. The truncation error is modeled by a uniform distribution and is added onto the measured acceleration which is integrated twice to calculate the position vector. Therefore, the truncation error on the accelerometer is integrated twice to determine the positional error. The error calculated from the integrations is considered to be the bias errors  $\epsilon_{\theta,bias}$  and  $\epsilon_{\phi,bias}$ . The apparent LOS is created using the calculated errors for the simulations via

$$\begin{aligned}\theta_{apparent} &= \theta_{true} + \epsilon_{\theta,nois} + \epsilon_{\theta,bias}, \\ \phi_{apparent} &= \phi_{true} + \epsilon_{\phi,nois} + \epsilon_{\phi,bias}.\end{aligned}\tag{53}$$

The final step in describing the error is randomly generating calibration times that occur less frequently than the sampling of the IMU. A simulation of the bias generated in MATLAB is depicted in Figure 16. The figure above shows a bias randomly changing over time similar to a random walk. Around the 150s mark, there is a noticeable



**Figure 16.** Bias simulation with added random noise from bias and noise with no measurements present

calibration to the error resulting in a value of zero radians of error before gradually accruing more error. A random noise was also added to the error in the figure to demonstrate what the final error being induced on the measurements will resemble. With the multiple measurements for the multiple sensors created, the next step is to apply the different methods of localization using the NLO for comparisons.

### 3.4 Procedures and Process

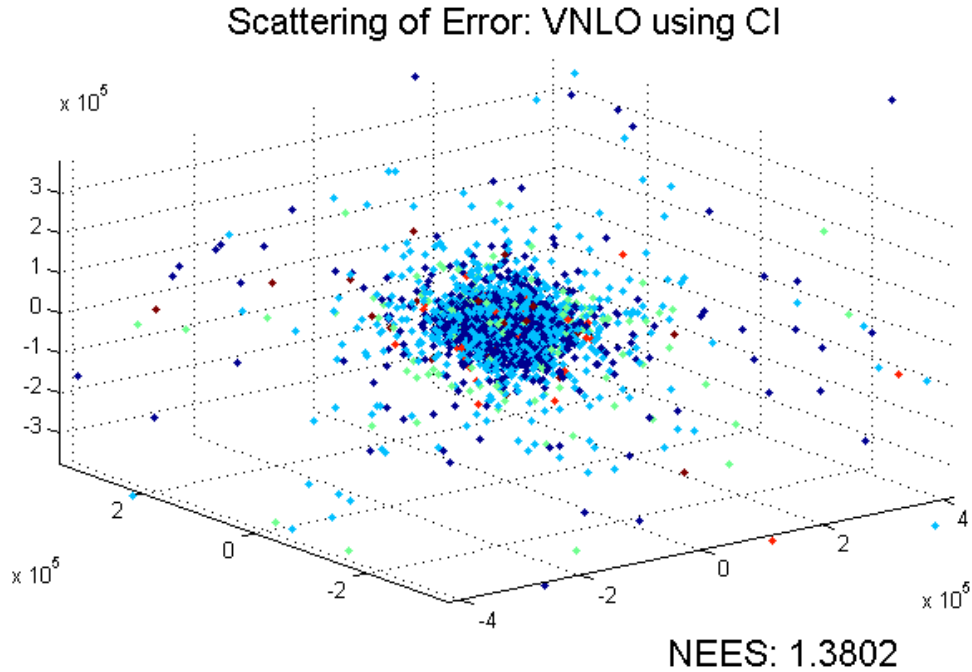
The analyzed NLO localization algorithms are:

1. Static NLO
2. Velocity NLO
3. Velocity NLO with CI
4. KM Velocity NLO with CI
5. KM Acceleration and Velocity NLO with CI.

The Velocity NLO is the benchmark localization method shown to be the most effective in the previous research for fast moving objects. The purpose of using the Velocity NLO is to depict how the error would increase once the simulations account for the correlated error by incorporating a bias. To account for the bias, the addition

of the CI algorithm and the KM are implemented. Finally, acceleration is added to the estimated state vector to investigate whether improves the ability to estimate an objects position. These algorithms are evaluated using Monte-Carlo simulations which allow for the calculation of the ANEES.

NEES, as the name suggests, is the error of the localization algorithm when compared to the true object path normalized based on the calculated covariance matrix. To determine whether a localization algorithm is over or under confident, the NEES will be greater than or less than the number of variables calculated by the algorithm. Because the state vector of the NLO will include position and velocity, the number of variables calculated is six. A sample MCS without the inclusion of bias error is shown in Figure 17.

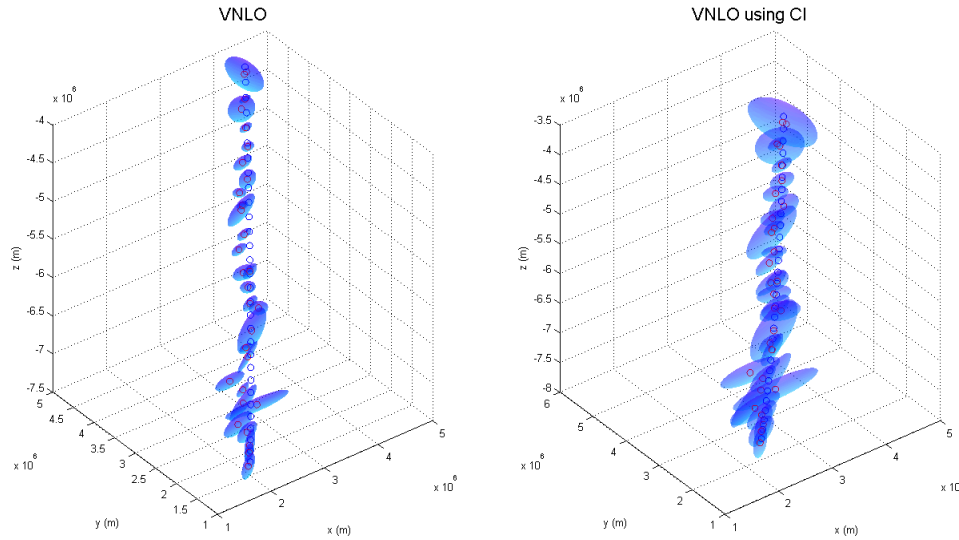


**Figure 17. Monte-Carlo simulation without simulating bias error**

The simulation in Figure 17 shows an example scattering of points centered on the origin when no bias is present in the measurement error. The locations represent the differences in the estimated and true object location. The color of the point

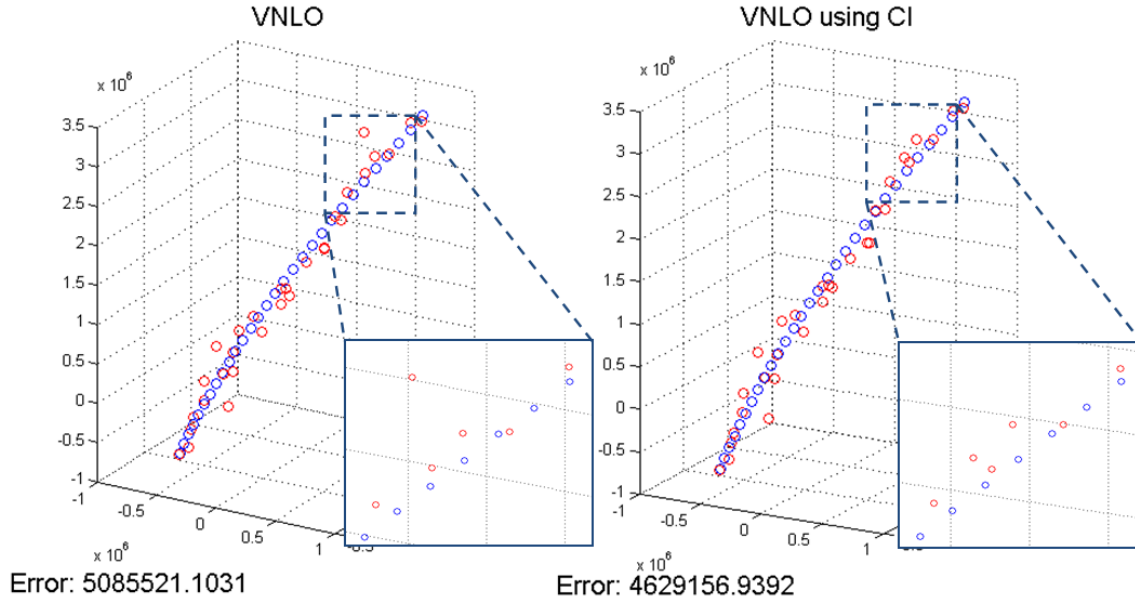
represents the associated NEES value of the estimation, where dark blue is an NEES value less than one, light blue is a value less than three, green is a value less than six, red is a value less than ten, and dark red is a value greater than ten. The average NEES value is shown in the bottom right-hand corner. The NEES value does not always represent more error as shown by red values near the center and blue values on the edges. The NEES value indicates the error in the estimation in conjunction with the confidence in the estimation. The larger the covariance ellipsoid for a given estimation, the more allowable error there is for an estimate. A larger covariance ellipse signifies an estimate calculated from measurements with less information. The larger ellipsoids are due to measurements from only a few sensors, or measurements taken from a similar location in space from multiple sensors. Therefore, an accurate estimation is more difficult to calculate due to a lack of information.

CI is expected to create larger error surfaces when coupled with the NLO algorithm. The intention for the addition is to create covariance ellipsoids without over-confident bounds. A simulation of this test is shown in Figure 18.



**Figure 18. Covariance ellipsoids along a path for the velocity non-linear optimization and velocity non-linear optimization using Covariance Intersection**

Figure 18 depicts the estimated path from the covariance intersection method is not significantly different from the velocity NLO without the covariance intersection method. However, the main difference when using covariance intersection is the increase in the covariance ellipsoid sizes encompassing the estimation points. Figure 19 shows the estimated paths without the depiction of the covariance ellipsoids.



**Figure 19.** True object paths (blue) and estimation paths (red) of the velocity non-linear optimization and velocity non-linear optimization using Covariance Intersection with associated total calculated error for the two different paths in meters

The error numbers shown on the bottom of the graphs in Figure 19 represent the total error in meters between the estimated and true estimation points without being normalized. The amount of error in a path is dependent on the amount of error exposed into a system. Therefore, the total error in the simulation is arbitrary, and the only significance of showing the values is to demonstrate that error is reduced when using the CI algorithm for this case. Some of the outliers in the path were corrected using the covariance intersection, but overall the error is not expected to be greatly reduced.

The second addition to the NLO algorithm is the use of a KM with the NLO. The purpose of the KM Velocity NLO CI is to address the issue of outlying estimations on an object path. Earlier versions of the velocity NLO estimate one point at a time rather than the entire path. The focus with the KM VNLO CI is to estimate the entire path with a single iterative loop. Estimating the entire path at once allows the KM Velocity NLO CI to optimize over the entire path rather than a single point. The KM Velocity NLO CI requires large matrices that are more computationally intensive. The computation time is greatly reduced with the use of sparse matrices in MATLAB. However, the research is post-processing, the time it takes to run the algorithms will be ignored.

### 3.5 Conclusion

The geolocation algorithms are based on NLO. Different algorithms estimate different state vectors of the same object of interest. The different state vectors include a combination of position, velocity, and acceleration. The main contribution of this research is the addition of the CI algorithm to the NLO algorithms. CI has been adapted to the NLO algorithms in the scenario where the amount of correlation among sensor measurements is unknown. In this scenario, we assume that the measurements are perfectly correlated providing a conservative or under-confident estimate of the covariance matrix. Covariance ellipsoids are produced from the covariance matrix to visualize confidence in an estimated position. Chapter IV evaluates the performance of the different algorithms in different scenarios; providing insight to the performance gains of the additions to the NLO algorithms.

The NLO algorithms are tested using MCS. The MCS allow for the ability to calculate the ANEES for each algorithm in each scenario. The different tests cover added uncertainty to the measurements, different object paths and object path dy-

namics, different uses of the algorithms with respect to the iterations performed, and window sizes used to produce the estimations. For the general tests of the different algorithms, the covariance ellipsoid sizes along with the MSE for the position estimates are evaluated. The NEES is an amalgamation of the two metrics. When the NEES is too high or low, we can examine whether it was due to a poor position estimate or a poor covariance estimate.

CI assumes that measurements are perfectly correlated. The covariance matrix produced from this algorithm is expected to produce lower values of the NEES by increasing the ellipsoid sizes. The ANEES is expected to be less than the desired ANEES of three when estimating positions. The estimates from the algorithm are classified as under-confident because the covariance ellipsoids include more error than expected from the sensors.

The addition of the kinematic model is expected to smooth outlying estimations that are contrary to the overall object path. Outlying estimations may be due to minimal sensor measurements, or measurements that come from only a few sensors. Diversifying the sources of measurements used to produce an estimate also improves estimate accuracy in this scenario, but this may not always be possible.

Finally, the addition of acceleration estimates within the state vector increases the ability to describe an object within a window of measurements. Most benefits using the Acceleration and Velocity NLO are expected to be realized when increasing the window size used to produce individual estimates. By increasing the window sizes, more measurements are used to produce estimated states. The more measurements used to produce an estimated state, the larger the time gap is between the estimated time and the measurement times. Therefore, if the velocity is changing over time, the less we can assume that within a larger window that the velocity differences are negligible between the estimated state and the state of the object at the measurement

times.

## IV. Results

### 4.1 Introduction

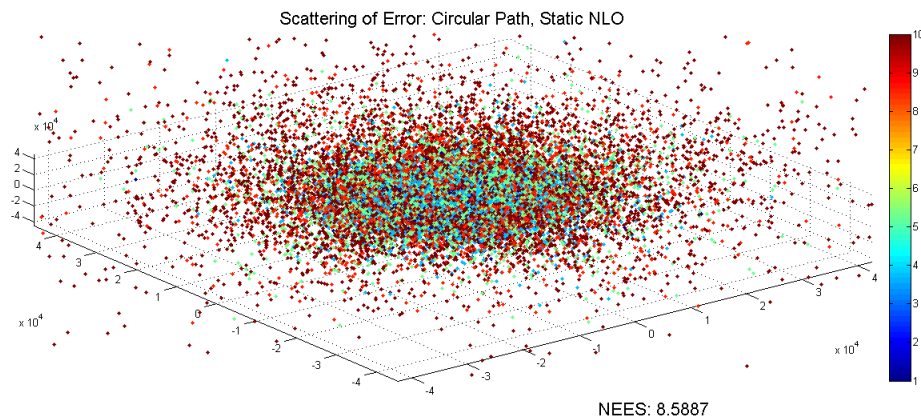
In this section, the results and the analysis for the results are discussed. Testing is performed on multiple algorithms and compared to the described triangulation algorithm. The first algorithm discussed is the Static NLO, which assumes that the non-time coincident data are generated from a stationary object, and thus, it is inferred that all LOS measurements are referenced to the same position in space and independent of their corresponding time stamp. Next, the Velocity NLO is analyzed which assumes that an object may be travelling at a constant velocity; therefore, within a window of time, the measurements used to find a single estimated point are known to be pointing at different positions along an object's path. Next, the CI algorithm is implemented and the effects on the estimated position and covariance matrix from the NLO are visualized for the instance when measurements are assumed to be correlated when taken from a single platform. A kinematic form of the NLO is also tested to analyze whether optimizing the NLO over an entire path with correlated measurements affects the estimated states. The variances for the kinematic model are based on the maximum values of an object's position, velocity, and acceleration. Finally, an Acceleration and Velocity NLO is formulated and tested to determine if there is any impact on the position estimates when the velocity of an object is no longer assumed constant. Two scenarios have been simulated to test the algorithms. Simplistic object paths of earth orbiting objects with random altitudes within LEO and inclinations are setup to determine how well the algorithms perform localization of non-dynamic objects. The simulations allow for the ability to vary the speeds of the object which effectively deteriorates the ability to localize for the algorithms that do not assume a kinetic object. Other than the simplistic path, a staged path is

generated such that an object has multiple stages at which forces acting upon the object change.

All results are be evaluated using three main criteria. The first two criteria are the MSE and ellipsoid sizes of the estimated positions and covariance matrices. The MSE of the estimated position evaluates the effectiveness of the algorithms at producing the most accurate position estimations. The ellipsoid sizes correspond to the MSE such that the smaller the MSE for a given algorithm, the smaller we would expect the ellipsoid sizes to be. The final criterion for evaluating the algorithms is the NEES values. In order to determine the expected value of the NEES, which is referred to as the ANEES, a MCS is used. The NEES is the most direct way of determining whether the expected error in the estimations is properly represented by the calculated state covariance matrices.

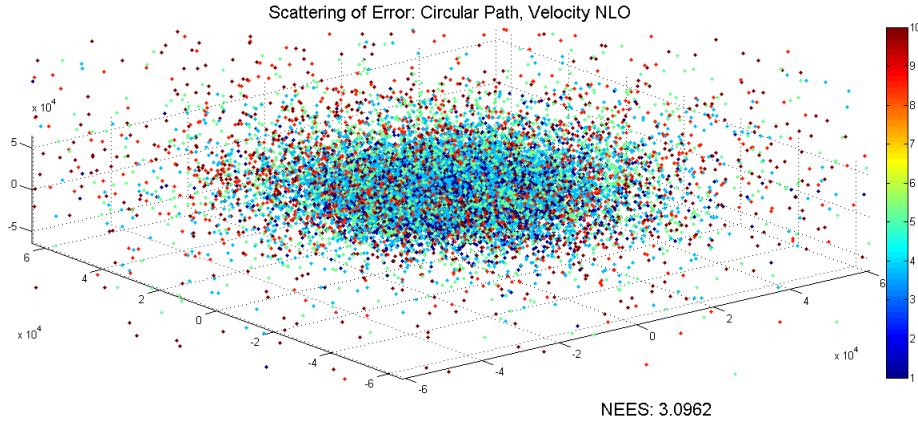
## 4.2 Results

MCS are used to analyze the estimations of the algorithms. Circular and complex paths for the object are used, and the scattering is shown in Figure 20.



**Figure 20.** Monte-Carlo simulation of Static Non-Linear Optimization used on a circular path of an object

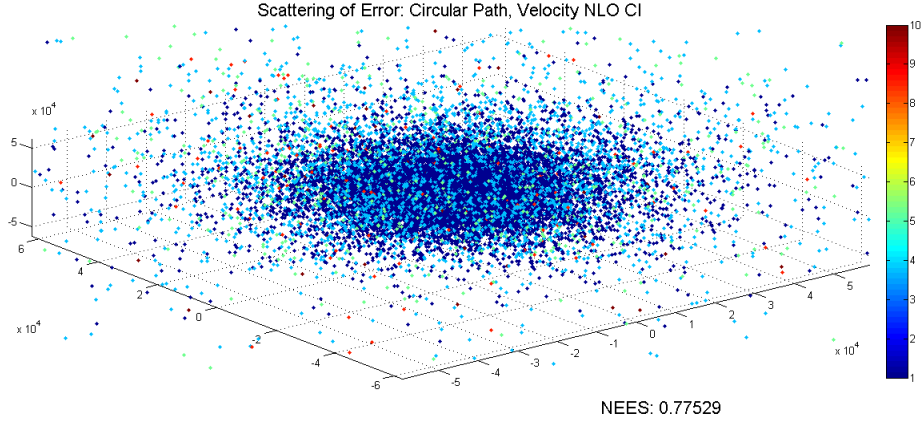
Using color to depict the NEES values, the individual estimations reveal that the difference between the true and estimated values directly influences the NEES. This is shown by the tighter grouping of blue points near the center of the scatter. The resulting average NEES for the MCS in Figure 20 is shown in the bottom right-hand corner of the Figure. For the static NLO used on a circular path, the NEES value is 8.5887, higher than the desired NEES. As described in equation 28, the desired NEES for the scenario is three because of the three degrees of freedom in a position estimation  $x$ ,  $y$ , and  $z$ . The next algorithm tested on the circular path MCS is the velocity NLO in Figure 21. The MCS for the Velocity NLO depicts a



**Figure 21.** Monte-Carlo simulation of Velocity Non-Linear Optimization used on a circular object path

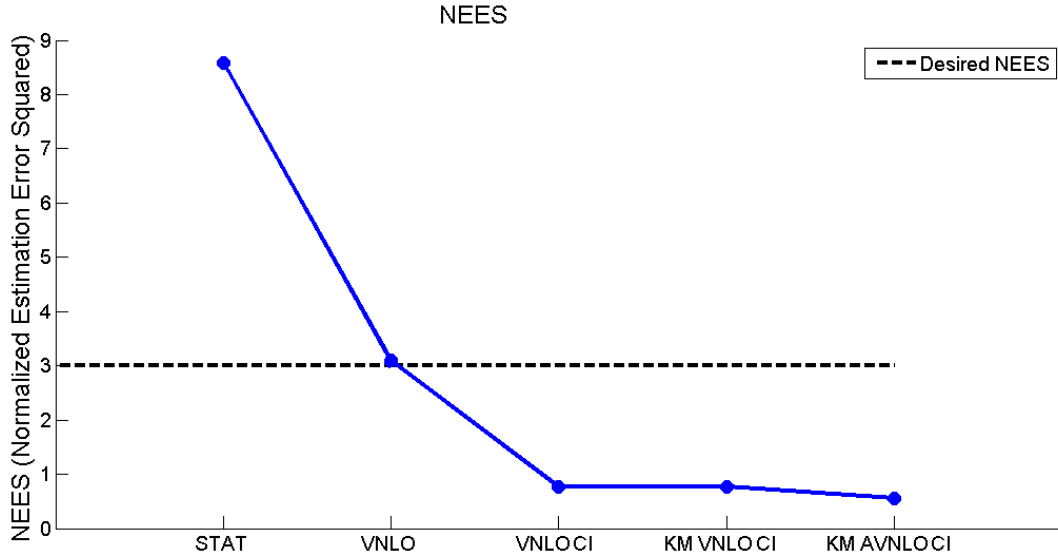
distribution that is centered around zero error. The majority of the points is colored in blue representing a NEES less than or equal to the value three, and the ANEES is just above three which is the desired value. Based on the information provided, the algorithm has shown to be a good estimator for the given scenario. If the NEES based on the metric from equation 28 has been met, then the CI algorithm is unneeded to produce better estimates. Figure 22 shows the effects of the CI algorithm on the circular path MCS.

The CI algorithm when used with the Velocity NLO produces a NEES value much



**Figure 22.** Monte-Carlo simulation of Velocity Non-Linear Optimization with Covariance Intersection used on a circular object path

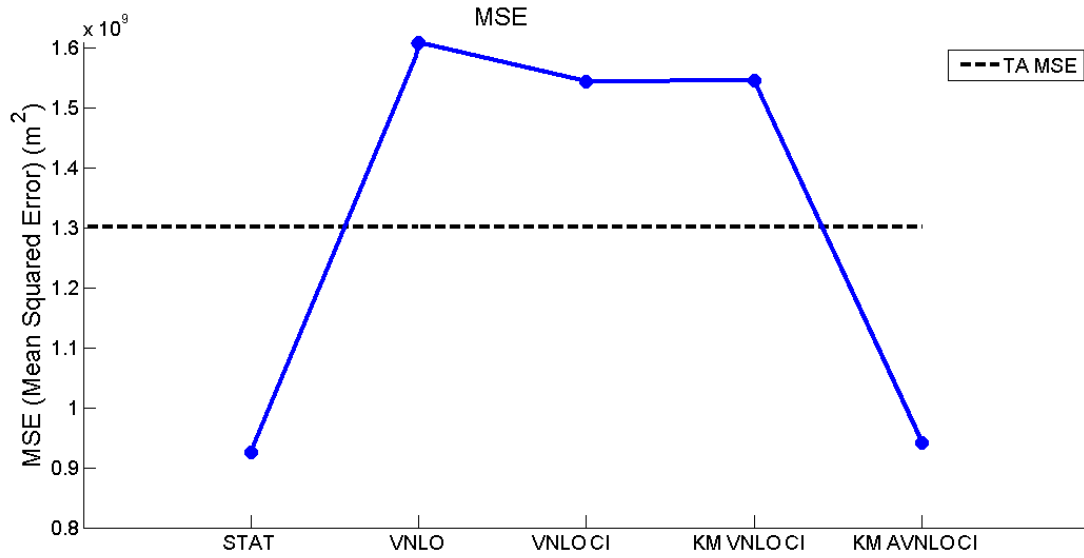
less than the metric of three. This implies that the assumption of measurements from a single sensor being perfectly correlated estimates a covariance matrix that is under-confident with respect to the positional error of the estimate. Therefore, based on the NEES, the Velocity NLO produces the best estimates of the position estimates and the corresponding covariance matrices. Figure 23 graphically depicts the decrease in NEES as more complexity is added to the NLO algorithm.



**Figure 23.** NEES values for the different algorithms on a circular path

The KM Acceleration and Velocity NLO are also included to show their effects on the NEES values. Using the circular path, the KM does not influence the NEES results significantly. The final point representing the acceleration and Velocity NLO shows a small decrease to the NEES value.

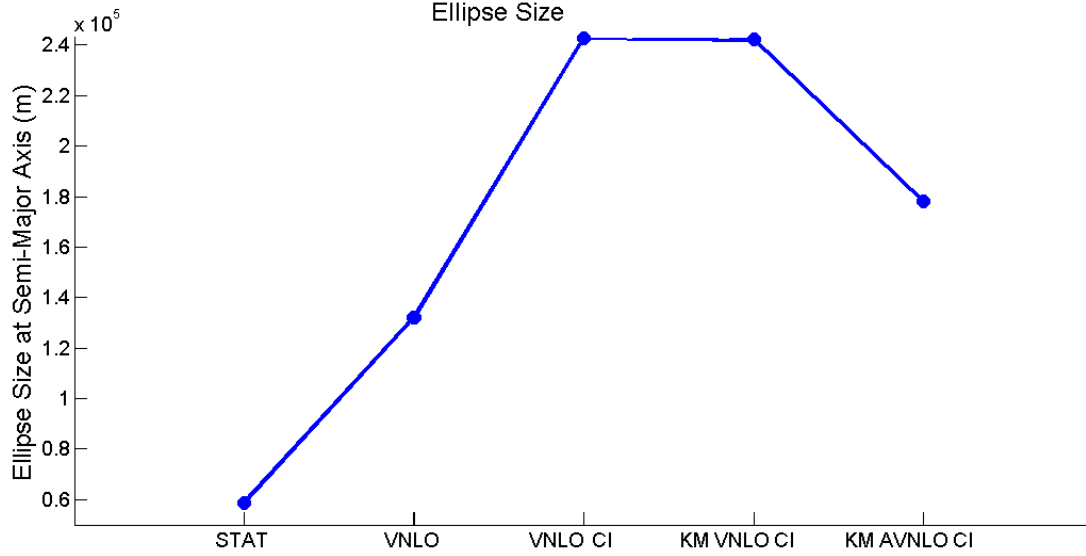
To fully understand the performance of the algorithms, the MSE and ellipsoid sizes are also investigated. The MSE is compared to the MSE of the original Triangulation Algorithm (TA) as shown in Figure 24.



**Figure 24.** MSE for the different algorithms on a circular path

Based on the MSE, only the static NLO and the Acceleration and Velocity NLO outperformed the TA.

The final metric of analysis for the NLO algorithms is the average ellipsoid sizes produced. Figure 25 shows the change in the semi-major axis of the covariance ellipsoids using different NLO algorithms where a smaller ellipsoid is desired.

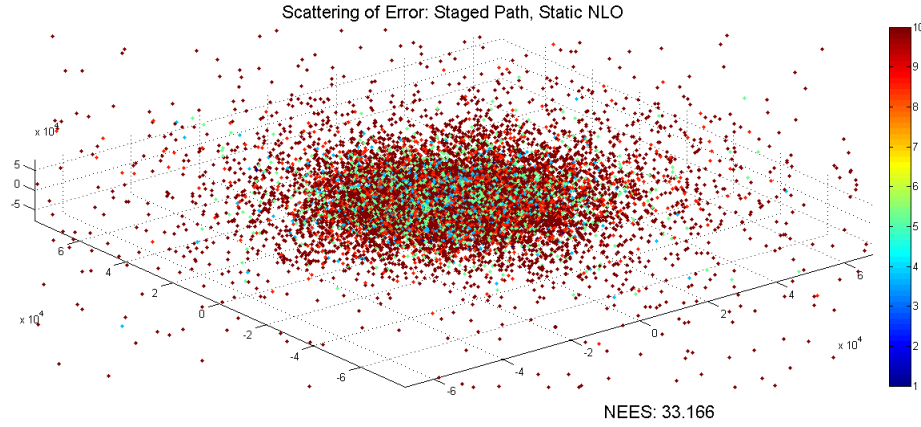


**Figure 25.** Average ellipsoid sizes for the different algorithms on a circular path

The largest change in ellipsoid size occurs with the addition of the CI algorithm. In the circular path simulations, the additional size is not needed to achieve the desired NEES value of three. The smallest ellipsoids are produced from the static case of the NLO. Because the NEES value for this algorithm shows that the algorithm was overconfident, the covariance ellipsoid is determined to be too small for the amount of error that was actually present in the system. However, the Velocity NLO produced the smallest ellipsoids for the algorithms that produced acceptable confidence values with a NEES value of 3.096 shown in Figure 23. In these aspects, the Velocity NLO produced the best results. Looking at the MSE, the Acceleration and Velocity NLO produced the best results.

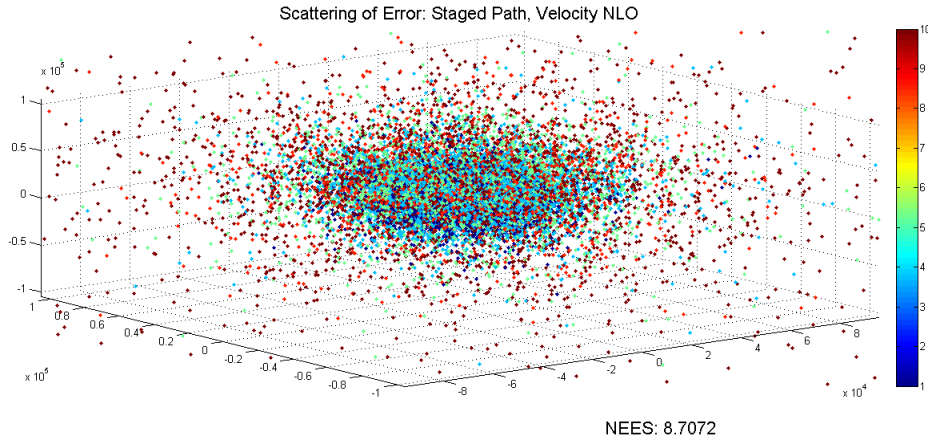
The next set of MCS simulations describe an object based on multiple stages an object may have. Figure 26 demonstrates the ability for the static NLO to produce estimates on these simulations

With a dynamic object path, the Static NLO results are dramatically higher in terms of NEES. The NEES for the Static NLO jumps from a value of 8.5887 in the



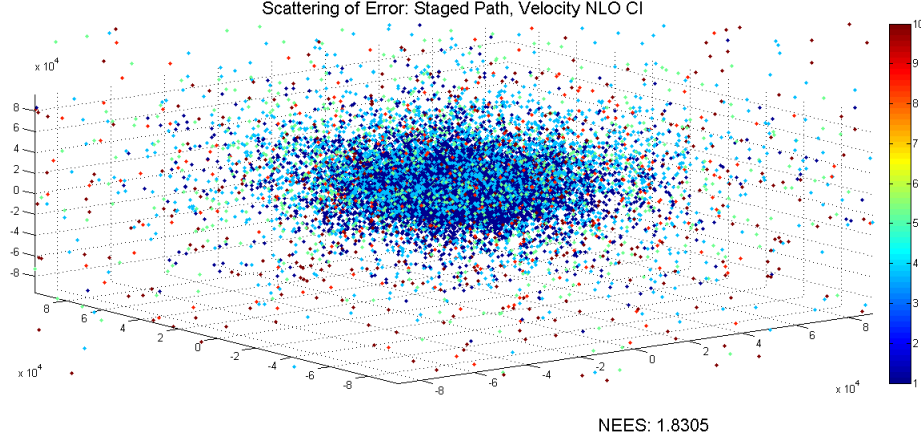
**Figure 26.** Monte-Carlo simulation of the Static Non-Linear Optimization used on a complex object path

circular simulations to a value of 33.166. An increase in the NEES is expected from the Static NLO because it assumes an object is stationary. Next, Figure 27 shows the results of the Velocity NLO against the staged path.



**Figure 27.** Monte-Carlo simulation of the velocity non-linear optimization used on a complex object path

The NEES for the Velocity NLO increases for the staged path similar to the Static NLO. A circular path the Velocity NLO produced the best results, but with the staged path, the Velocity NLO's performance degraded to that of the Static NLO used on the circular path. The next simulation evaluates the CI algorithm to determine if it can produce useful estimates with the staged path.

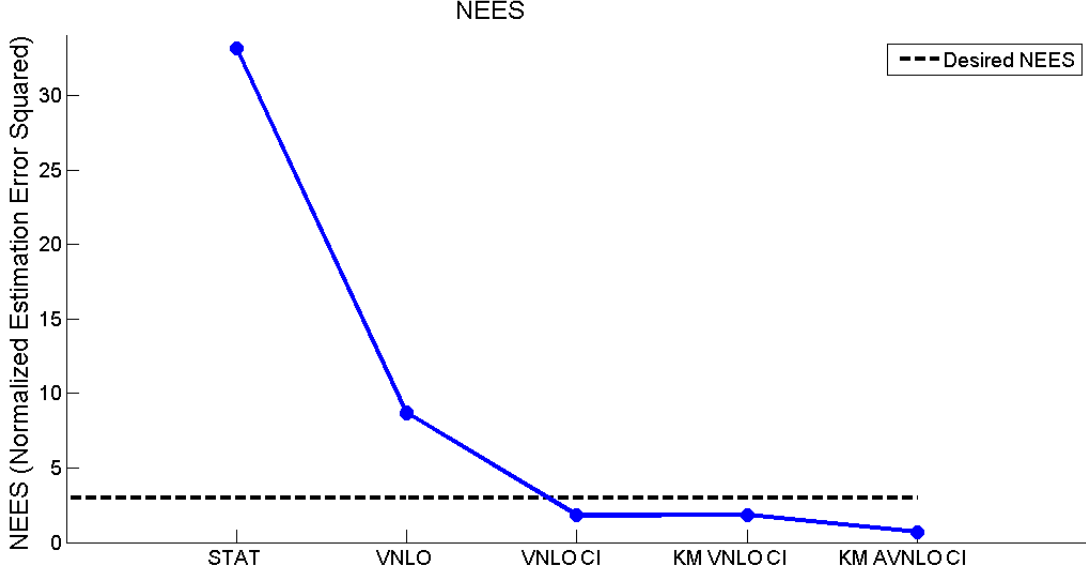


**Figure 28.** Monte-Carlo simulation of the Velocity Non-Linear Optimization with Covariance Intersection used on a complex object path

Using the CI algorithm with the NLO, the NEES drops to an acceptable level of 1.8305. The NEES in this case is once again less than the desired value of three. When evaluating the NEES it is preferable to be under-confident resulting in a NEES less than the desired value of three. The true covariance ellipsoid in the under-confident circumstance is a subset of the calculated covariance ellipsoid; therefore, the true error is accounted for. This is the expected outcome using the CI algorithm as described in Chapter II. Using the two additional algorithms, a graph of the NEES values when using a staged path MCS is shown in Figure 29.

In Figure 29 the only algorithms to perform at or lower than the NEES metric are the algorithms including the CI algorithm. The addition of a transition matrix does not significantly change the NLO algorithms' performances with respect to the overall NEES value. The Acceleration and Velocity NLO decreases the NEES value as was the case with the simple path. In order to fully analyze the NEES results, the next step is to investigate the MSE values and ellipsoid sizes to determine how the NEES values were decreased. Figure 30 shows the MSE values for the MCS using a staged path.

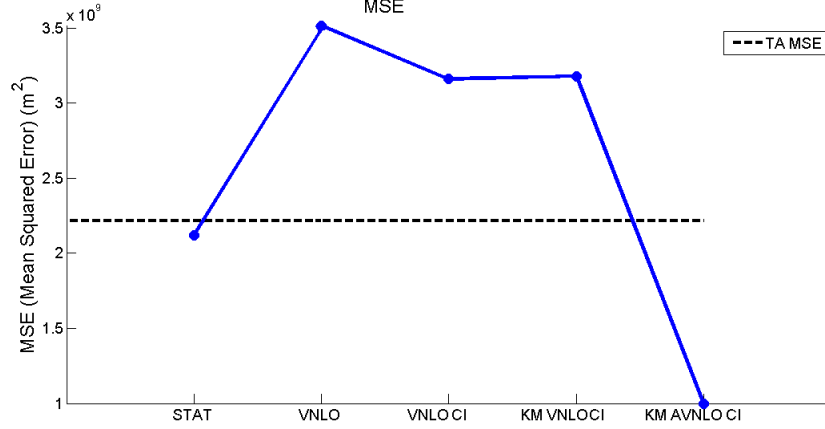
Using a staged path to evaluate the MCS, the MSE of all algorithms increases



**Figure 29. NEES values for the different algorithms on a staged path**

significantly with the exception of the Acceleration and Velocity NLO. Because the Acceleration and Velocity NLO takes into consideration the possibility for a changing position within a window, it is able to produce similar comparable results to the circular path case in Figure 24. The significantly smaller MSE when compared to the other algorithms validates the smallest NEES value seen in Figure 29. Figure 31 shows the change in the semi-major axis of the covariance ellipsoids for the different algorithms which inherently describes the size of the ellipsoids.

The ellipsoid sizes are consistent with the sizes seen in the circular path MCS. The consistency in the ellipsoid sizes is because in both sets of simulations, the same amount of information is used to produce the estimates. The ellipsoids are dependent on the amount of information used and not the dynamics of the object. Regarding equation 8, if the information matrix  $I(\mathbf{X})$  does not change, then the CRLB does not change, thus the same covariance ellipsoid size is expected. Therefore, with exception to the acceleration and velocity NLO, the increase in the NEES values is due to the degraded ability for the algorithms to produce accurate estimated values.

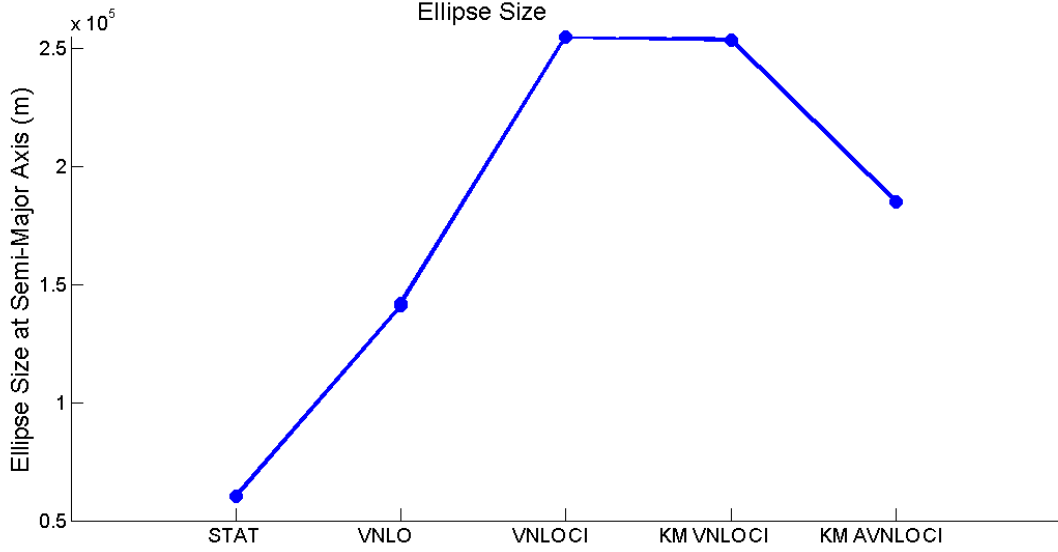


**Figure 30.** MSE for the different algorithms on a staged path

The next set of simulations analyze the Velocity NLO, Velocity NLO with CI, the KM of the Velocity NLO with CI, and finally the KM of the Acceleration and Velocity NLO with CI with varying parameters. The parameters that are of interest are the variance in time measurements, the residual on the iterations of the NLO algorithms, the speed of the object being localized, and finally the window size used to create the individual position estimates.

Figure 32 depicts the NEES against an increasing uncertainty in the measurement times. Adding small variations on the timing measurements simulates a scenario where an unknown error is present in a system that is not taken into account by the localization algorithms. The added error to the measurement times ranges from zero to fifty milliseconds is shown in Figure 32.

Most of the degradation to the algorithms occurs when small amounts of timing error are first introduced to the system. The velocity NLO with CI is nearly identical to the KM velocity NLO with CI in this simulation along with the rest of the simulations. When the CI is adapted to the NLO, the amount of degradation to the NEES for the timing error is insignificant when compared to the amount of error assumed with correlated errors.



**Figure 31. Average ellipsoid sizes for the different algorithms on a staged path**

The next simulation analyzes how many iterations of the NLO are necessary to converge onto a optimal estimate. The larger the residuals, the less iterations of the algorithm that are performed. After one iteration, the rate of convergence to a given residual is dependent on the path of the object. The results for the simulation are shown in Figure 33.

When evaluating the algorithm based on the NEES, a single iteration is sufficient to produce state and covariance matrices with values similar to fifty iterations. The NLO more dependable converges onto a solution when given a good initial estimate which for this research is given by the TA algorithm. The NLO takes the initial estimate and fine tunes the state to better agree with the measurements; therefore, only a few iterations are needed to get the final estimate. This incident has been shown in prior literature [15]. The Acceleration and Velocity NLO in particular exhibits little change in the NEES over multiple iterations. Figure 33 shows that the NEES remains slightly above a value of 0.5 for all residual values.

The next simulation analyzes the algorithms against an object at different speeds.

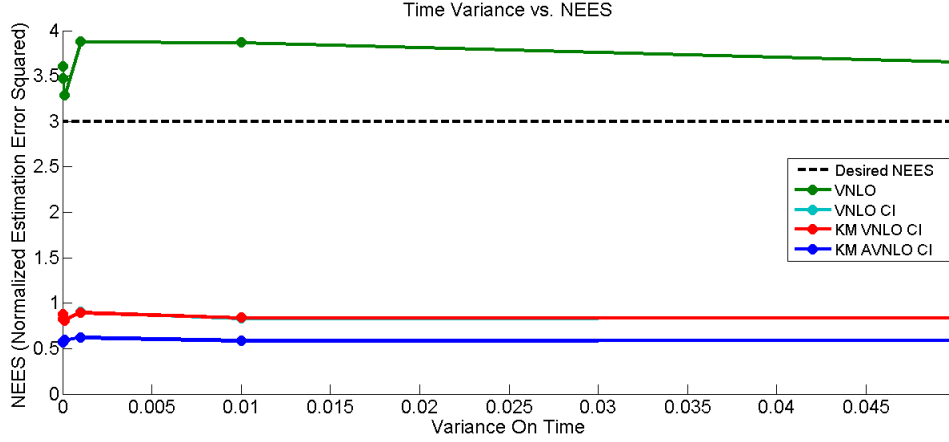


Figure 32. Time variance testing on the different algorithms for a circular path

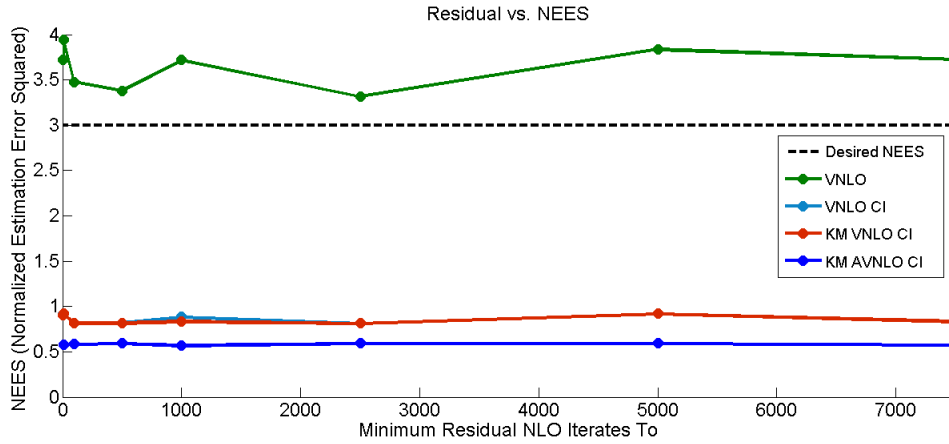


Figure 33. Residual testing on the different algorithms for a circular path

A circular path is used so the speed of the object stays constant during its path of motion. Therefore, the localized object does not exhibit any acceleration except when examining the individual  $x$ ,  $y$ , and  $z$  values. The results are shown in Figure 34

A commonality among all algorithms is the divergence that occurs as the object of interest increases in speed. The difference in the algorithms is the rate at which they diverge from their original NEES value. When an object is traveling slower than 2500 m/s, the Velocity NLO with CI has a lower NEES value than the Acceleration and Velocity NLO. The Acceleration and Velocity NLO diverges more slowly than any of the other NLO algorithms. Therefore, the speed of the object of interest dictates the

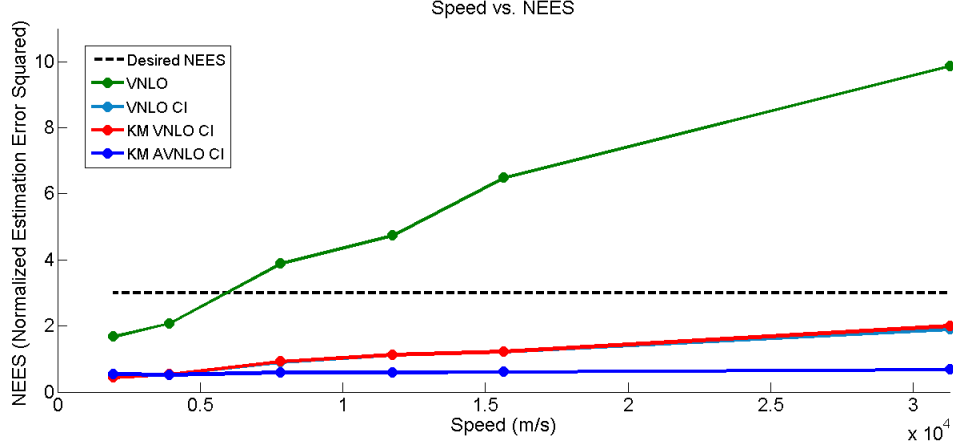


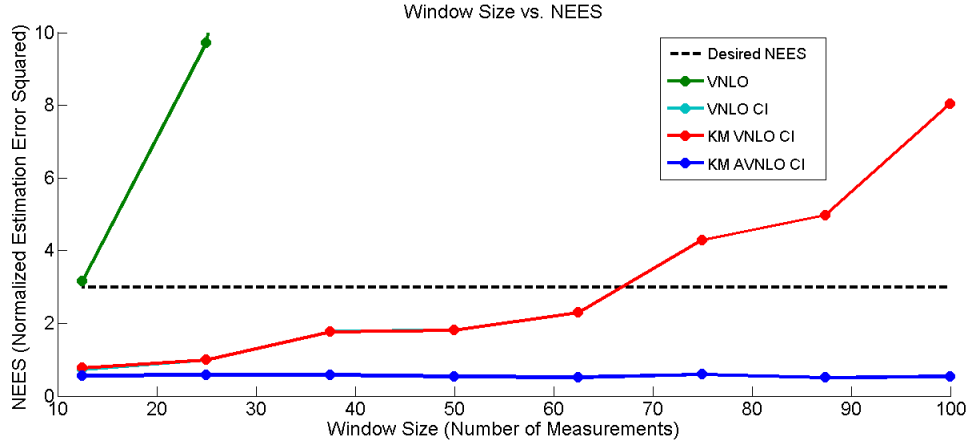
Figure 34. Speed testing on the different algorithms for a circular path

effectiveness of the different algorithms.

The final simulation assesses the algorithms on their ability to handle larger window sizes. Window size is dictated by the number of measurements that are being used to calculate an estimation point. The measurements are allocated over time based on a uniformly random distribution. By increasing the number of measurements, the size of the window increases to reach the number of measurements needed to create an estimate. The greater number of measurements also increases the amount of information for a single estimate which decreases the size of the expected covariance ellipsoid. Because the measurements are also further separated, assumptions about the object within a window of data are no longer valid. As an example, for a dynamic object, an object's path over a small window in time can be approximated with a constant velocity. As the window increases, it becomes necessary to create a better approximation of the object's path.

The results of testing different window sizes is similar to the speed tests. Increasing the window size increases the distance the outlying measurements are from the estimated position. Increasing the speed of the object disperses the object's positions in space over a given window in time. Increasing the window size increases the number of measurements, but also increases the change in position between the first and

last position. Therefore, a similar phenomenon is seen in Figure 35 as with the speed test.

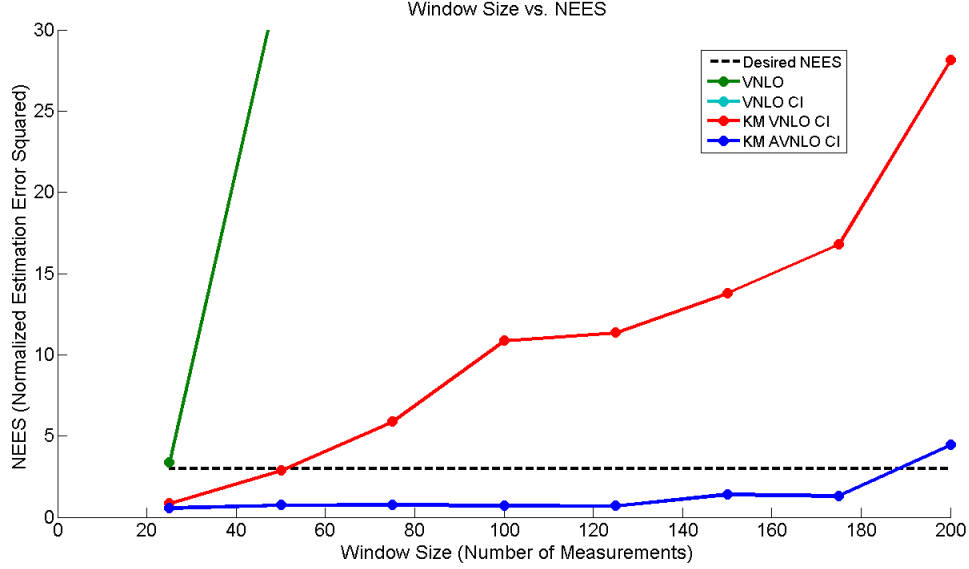


**Figure 35. Window testing on the different algorithms for a circular path**

The NEES value of the velocity NLO diverges much more quickly than the other algorithms. However, the Acceleration and Velocity NLO's NEES values do not change based on the window size. The algorithm is able to properly describe the object's motion within the window, thus creating better estimates when more data is present despite the fact the window size increases. In order to show that the Acceleration and Velocity NLO in fact degrades in ability to estimate with larger windows, Figure 36 depicts what happens when a window increases to a size large enough to encompasses multiple stages of an object.

In Figure 36, the NEES value of the AVNLO holds its value much better than the other algorithms, producing under-confident covariance matrices for the largest window sizes. Here, a another stage falls into the window creating the sudden divergence past the desired NEES value.

Achieving a NEES value of three means that the estimate from the algorithm lies within the 95% probability region of the estimated covariance matrix. Therefore, we expect the majority of the estimates to fall within the 95% probability region when



**Figure 36. Windowing testing on the different algorithms for a complex path**

plotting the individual NEES values over time for a circular path shown in Figure 37.

Because the NEES values follow a Chi-Squared distribution, the 95% probability region for an estimator with three degrees of freedom is

$$\chi_3^2(0.95) = 9. \quad (54)$$

The Velocity NLO has the closest ANEES value to three when using a circular path. Figure 37 shows that roughly six out of one hundred estimates were greater than nine, or 94% of estimates fell within the region validating the claim in equation 54. When using a staged path to evaluate the algorithms, the resulting NEES from the algorithms tend to have higher and more sporadic values as depicted in Figure 38.

For the Velocity NLO CI algorithm and the KM velocity NLO CI algorithm, the ANEES values are well below three, but the NEES values between 150 and 160 are much higher than nine. The reason for the outlying NEES values is most likely due to a change in stages in the object at this point in time. Because all other values of the NEES for the simulation are much lower, an ANEES much lower than three was

achieved.

### 4.3 Conclusion

The base NLO algorithm with various adaptations has been tested under different conditions. The NLO algorithms were tested and the ability to create position estimates were evaluated by calculating the MSE, ellipsoid sizes, and ANEES.

The base NLO algorithm known as the Static NLO consistently produced the highest values of the NEES. The NEES value provides a useful metric to analyze the algorithms because the NEES values described how well the calculated covariance matrix agreed with the true position error. Because the static NLO was sub-optimal at estimating an object's state, we used Velocity NLO to produce more accurate estimates. The Velocity NLO produced better NEES values in certain conditions, but was proven not to be consistent with a staged path. Once the CI algorithm was applied to the NLO algorithms, the NEES values were much more consistent under different scenarios at the cost of increasing the ellipsoid sizes. The acceleration and velocity NLO using the CI algorithm produced the best overall results when using all scenarios, and consistently produced the lowest MSE of all algorithms. The KM addition to the algorithms did not produce significant benefits when compared to the algorithms without the KM. The model was used with large variances relating to the largest acceleration values that may be exhibited by an object. Because the variances were so large, little if any change actually occurred within the estimations. The simulations were based on object paths, different object speeds, uncertainty within the system, the number of iterations, and window sizes of the algorithms.

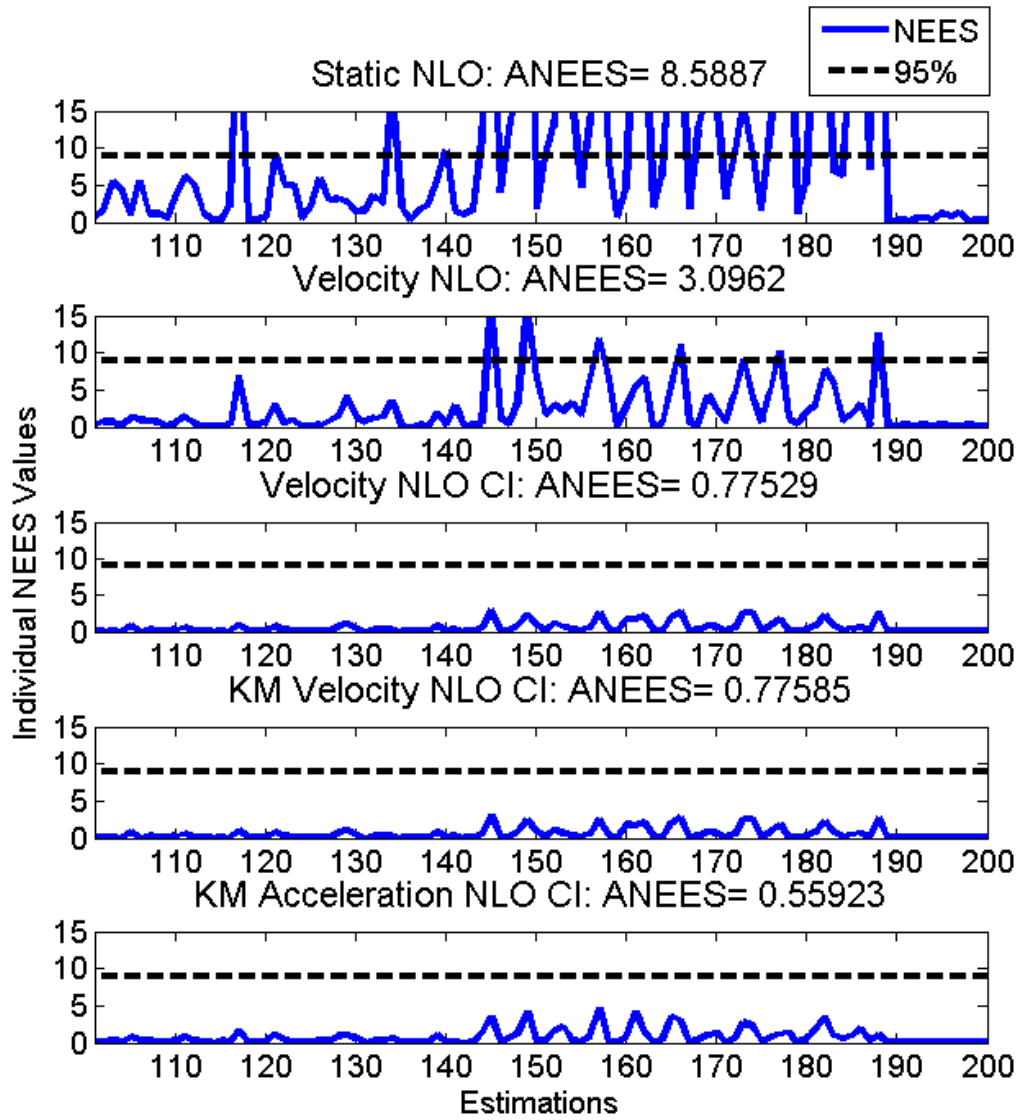


Figure 37. NEES values for multiple estimation points occurring over time on a circular path

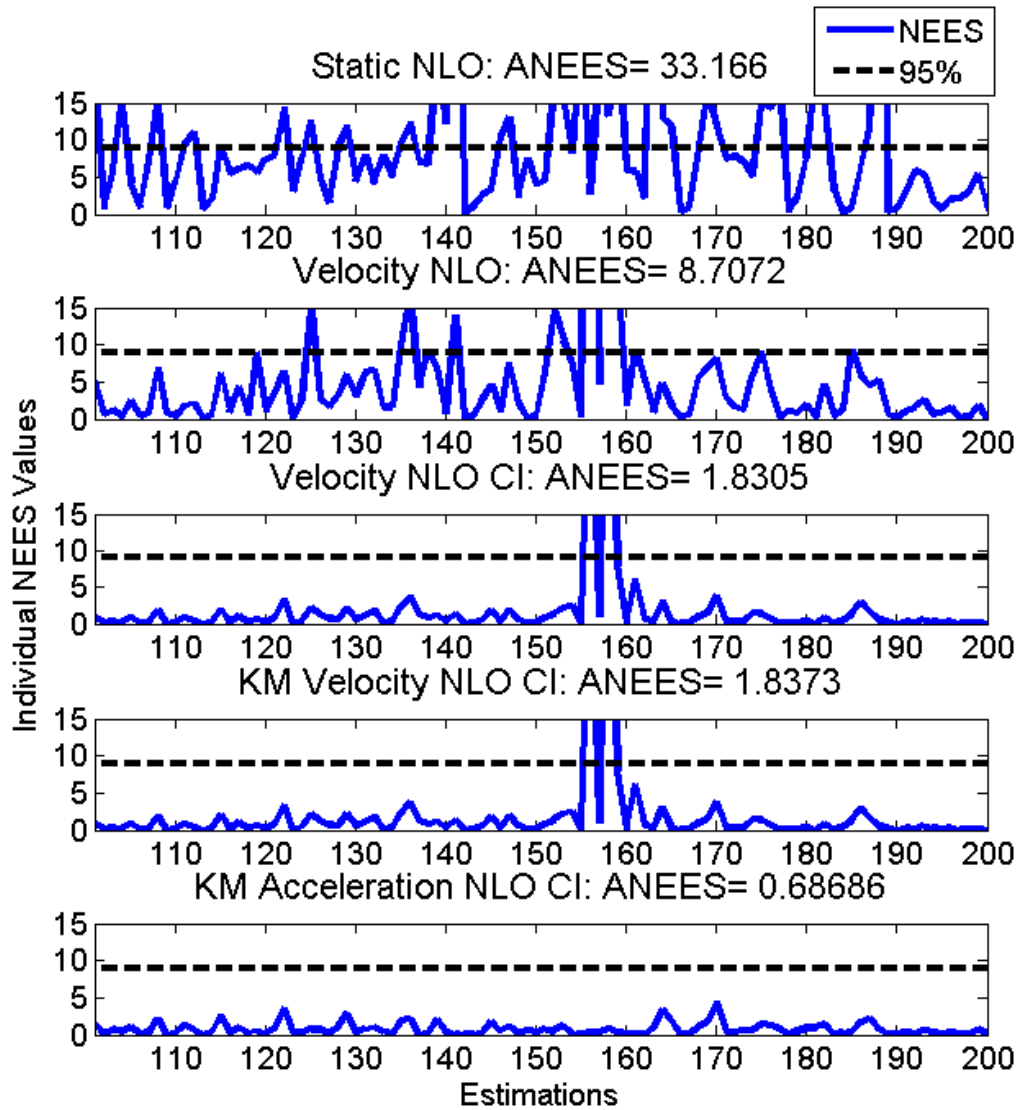


Figure 38. NEES values for multiple estimation points occurring over time on a staged path

## V. Conclusion

### 5.1 Introduction

In this chapter, the final analysis of the results from chapter IV are discussed. MCS were used to produce scatterings of estimation errors which were used to produce values for the ANEES. The ANEES values were used to produce general plots of the different NLO algorithms to depict how well each algorithm was able to create estimates. The algorithms used were the Static NLO, Velocity NLO, Velocity NLO using the CI algorithm, the KM Velocity NLO using the CI algorithm, and finally the KM Acceleration and Velocity NLO using the CI algorithm. Different scenarios were used to determine how the different algorithms were able to perform in the different circumstances.

Future work for the research is discussed in this chapter with focus on the initial testing of the Expectation Maximization algorithm. The purpose of Expectation Maximization is to create likelihood functions of the different stages in an object's path.

### 5.2 Simulations

The first set of simulations analyzes the different NLO algorithms on a given set of parameters to find the general trends between the algorithms. All algorithms have the same amount of simulated error with the variance of the bias and noise being held constant.

When testing a circular path, each version of the NLO algorithm when adding complexity achieves progressively lower NEES values. It was determined that the Velocity NLO algorithm produced the expected ANEES value while the Static NLO algorithm had an ANEES value above the desired value and all of the other algorithms

produced lower ANEES values. The addition of the acceleration estimates within the state vector had the lowest MSE while producing covariance ellipsoids that were under-confident of the true covariance ellipsoid. Under-confidence signifies that the covariance matrices represent at a minimum the true amount of error in a system. Therefore, under-confidence is preferable to over-confidence.

Testing against a staged path produced different results when compared to the circular path tests. The Velocity NLO no longer produced the desired ANEES values as before, but was over-confident producing an ANEES similar to the Static NLO in the circular path simulations. The Acceleration and Velocity NLO saw little decline in performance between the two different simulated object paths.

The simulations also examined the possibility for errors from other sources. The tests using different variances on time revealed that all of the algorithms with CI had negligible effects on the NEES. The CI algorithm already produces a conservative estimate of error, so small sources of error were not able to cause significant deviations in the NEES. Tests on the amount of iterations of the NLO also showed minimal effects after the first few iterations of the NLO. The need for only a few iterations of the algorithm was described in [15] which states that an accurate initial estimate is necessary to converge; therefore, only a few iterations of the Gauss-Newton method are necessary to further improve localization accuracy. Our results confirm this statement.

The two analyses that produced the most significant results were the speed tests, and the windowing tests. When the speed of the object of interest increased, degradation to the estimation's accuracy in all algorithms with exception to the Acceleration and Velocity NLO was exhibited. Similar results were seen with the windowing tests. The Acceleration and Velocity NLO uses fewer assumptions about an object's path within a given window of time. Therefore, the Acceleration and Velocity NLO was

able to most accurately describe an object within a window of time. The KM Acceleration and Velocity NLO using CI is able to perform under the most conditions.

The CI algorithm consistently produced under-confident covariance matrices depicting error in the position estimates. When the CI algorithm was not used, it is possible to calculate covariance matrices that are accurate and produce smaller ellipses, but these algorithms were exposed to be inconsistent in different tests. Therefore, the CI algorithm is best used when the path and amount of correlated error is uncertain. Later work may look into the possibility of assuming only partial correlations between measurements rather than perfect correlation as in the tests for this research. Perfect correlation within error was assumed in these tests because we assumed the total amount of error is known, but the correlations in error were unknown other than the total amount of error assumed on the measurements.

Using a transition matrix to describe a KM over an entire path proved to create insignificant results when the variances used to describe confidence in the model were too large. The reason for using larger variance terms was to ensure the estimates were not being fit to a path. The assumption was that we were dealing with an unknown object, and the desired covariance matrices depicting error described the error from the sensors; therefore, this allows for the ability to analyze the potential of the given sensors to produce estimates. The KM therefore was used to ensure that viable estimations were being produced from the given sensor measurements. For the given simulations, the transition matrices did not exhibit any significance, but if the variances can be reasonably decreased the transition matrices may be shown to produce better results.

Adding acceleration to the estimated state vector was able to increase the accuracy of the position estimates as described by the calculated MSE values. The Acceleration and Velocity NLO provided consistent results in terms of the NEES. The drawback

of estimating acceleration is the increase in computational cost. Because of the more complex computations, the Jacobian matrix for the algorithm is calculated using estimated partial derivatives. The estimation method for the derivatives increases the error, but the increased error was unnoticed within the test results.

### 5.3 Conclusion

In conclusion, the multiple algorithms for producing geo-localization estimates performed differently depending on the scenario. In the circular path simulations, the CI algorithms consistently produced under-confident estimates of the covariance ellipsoids. The consistency may be desirable, but in some situations the Velocity NLO without the CI algorithm produced the best results with respect to the NEES values. However, the inclusion of the Acceleration and Velocity NLO produced the best results with respect to the MSE. The Acceleration and Velocity NLO consistently produced under-confident covariance ellipsoids with only a single exception from the tests. The exception occurred when single estimations were produced from windows including measurements of the multiple stages an object may have. Overall, position and covariance estimates can be produced for a geo-localization problem with measurements that have correlations due to biasing by using a NLO algorithm based on the Gauss-Newton method.

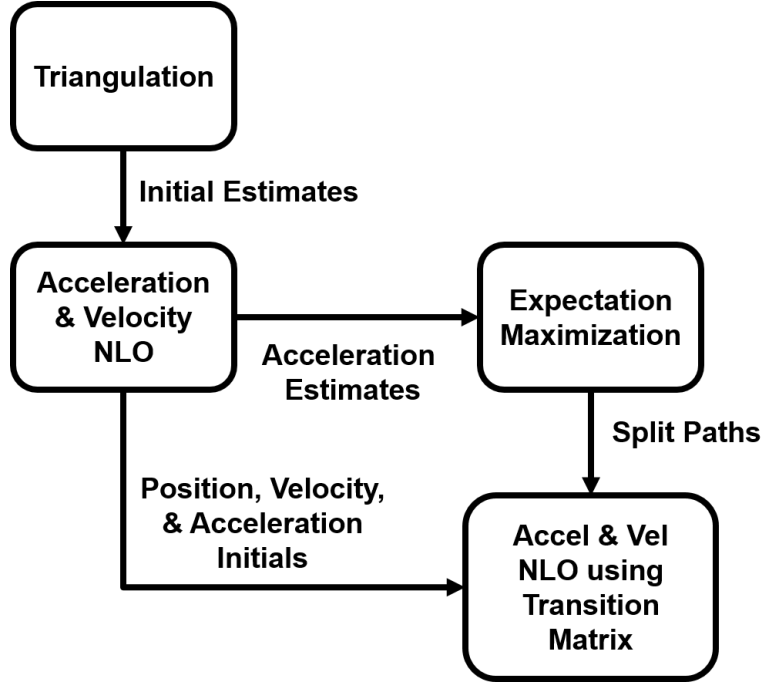
#### **Future Work.**

Future work on the subject may include partitioning measurements into multiple paths based on the stages an object can have over time. We were able to observe that when using a staged path, all of the geo-localization algorithms at some point are unable to produce accurate estimates when a window includes measurements of an object with multiple stages. Therefore, if a path can be divided into multiple smaller

paths this problem can be mitigated.

One method of dividing data into multiple groupings are through the use of multiple likelihood functions. A technique known as Expectation Maximization is used to iteratively augment multiple likelihood functions on a set of data to produce the maximum expected value that all data points fall within a set of given likelihood functions [32]. The assumption for the set of data is that the data is sampled from different distributions. Therefore, if we assume that all distributions are Gaussian, then on each iteration of the Expectation Maximization algorithm, the likelihood functions change their mean and variances to accommodate the different distributions composing the data.

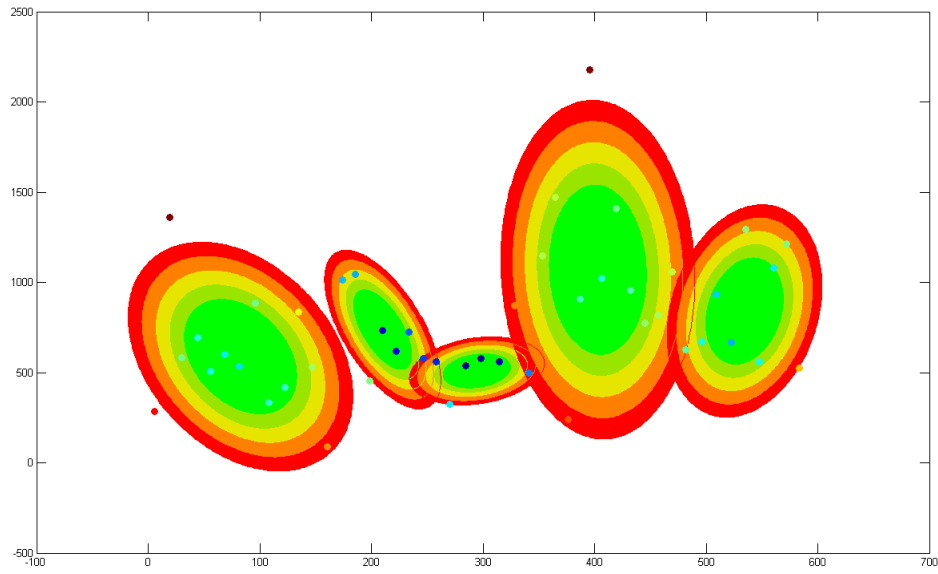
The idea for using the Expectation Maximization algorithm on the geo-localization problem is that the different stages have different acceleration values attributed to them. A stage is defined by a change in force on an object or a change in mass therefore using Newton's second law, we expect a change in acceleration on an object. Using an Acceleration and Velocity NLO, we are able to produce initial estimates of the acceleration that can be used with the Expectation Maximization algorithm. The likelihood functions being calculated by the algorithm will have a mean value associated to the mean of the acceleration over a given stage. A flowchart of the combination of algorithms is depicted in Figure 39.



**Figure 39.** Flowchart of NLO algorithm incorporating Expectation Maximization

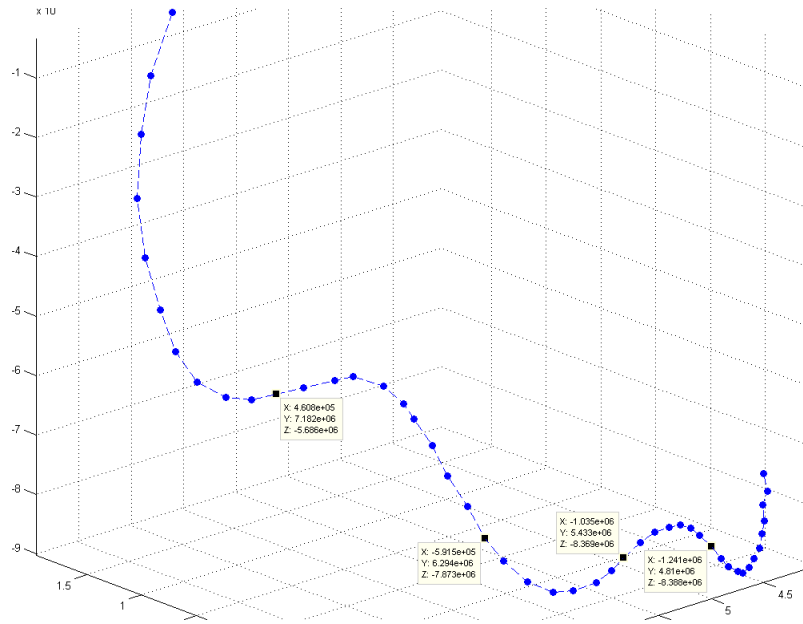
The variance values from the Expectation Maximization likelihood functions may also be used as the covariance term  $\Sigma_{trans}$  within the transition matrix. Because the variance terms used within the transition matrix are large, the KM does not significantly change the current estimates. However, if the lower variance terms from the Estimation Maximization algorithm are used, significant improvements to the estimations may be seen.

An initial test of the Expectation Maximization algorithm has been performed on the acceleration estimates to examine the ability for Expectation Maximization to partition a given object path as depicted in Figure 40. The likelihood functions in the figure are depicted by the ellipses of varying colors.



**Figure 40. Initial test of an Expectation Maximization algorithm on estimated acceleration values from a single path**

The test of the Expectation Maximization algorithm exhibits the ability to create likelihood functions from seemingly scattered acceleration values. The path of the object that was used to run the Expectation Maximization test is shown in Figure 41. The points at which a change in likelihood functions changes has been marked, and in this scenario, the Expectation Maximization algorithm produced seemingly accurate estimations.



**Figure 41.** Path of object used in initial Expectation Maximization test

Further analysis of the algorithm is needed to determine if Expectation Maximization can consistently produce accurate sub-paths. If the paths can be consistently produced, the next step would be to see if there is a significant impact on the ability to produce position and covariance estimates.

## Bibliography

1. S. Hartzell, “Non-linear Optimization Applied to Angle-of-Arrival Satellite-Based Geolocation,” M.S. thesis, Air Force Institute of Technology, 2014.
2. A. Kealy, G. Retscher, A. Hasnur-Rabiain, N. Alam, C. Toth, D.A. Grejner-Brzezinska, T. Moore, C. Hill, V. Gikas, C. Hide, C.; Danezis, L. Bonenberg, and G.W. Roberts, “Collaborative Navigation Field Trials with Different Sensor Platforms,” *IEEE 10th Workshop on Positioning Navigation and Communication (WPNC)*, pp. 1–6, 2013.
3. Y. Zhou, H. Leung, and M. Blanchette, “Sensor Alignment with Earth-Centered Earth-Fixed (ECEF) Coordinate System,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 35, no. 2, pp. 410–418, Apr. 1999.
4. G. Strang, *Linear Algebra and its Applications*, Brooks/Cole, Belmont, CA, 2006.
5. Jun Xu, Maode Ma, and C.L. Law, “AoA Cooperative Position Localization,” in *IEEE Global Telecommunications Conference*, Nov 2008, pp. 1–5.
6. Jehoshua Bruck, Jie Gao, and Anxiao Jiang, “Localization and Routing in Sensor Networks by Local Angle Information,” *ACM Trans. Sen. Netw.*, vol. 5, no. 1, pp. 7:1–7:31, Feb. 2009.
7. Tolga Eren, Walter Whiteley, and Peter N Belhumeur, “Using Angle of Arrival (Bearing) Information in Network Localization,” in *Decision and Control, 2006 45th IEEE Conference on*. IEEE, 2006, pp. 4676–4681.
8. Guangwei Zhu and Jianghai Hu, “Distributed Network Localization using Angle-of-Arrival Information Part I: Continuous-time protocol,” in *American Control Conference (ACC), 2013*, June 2013, pp. 1006–1011.

9. R. Prevost, M. Coulon, P. Paimblanc, J. LeMaitre, J.-P. Millerioux, and J.-Y. Tourneret, "Ship Localization Using AIS Signals Received by Satellites," in *Signal Processing Conference (EUSIPCO), 2013 Proceedings of the 21st European*, Sept. 2013, pp. 1–5.
10. Yang Zheng Bin, Wang Lei, Chen Pei Qun, and Lu An Nan, "Passive Satellite Localization Using TDoA/FDoA/AoA measurements," in *Conference Anthology, IEEE*, Jan 2013, pp. 1–5.
11. K. Ho and Y. Chan, "Geolocation of a Known Altitude Object from TDoA and FDoA Measurements," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, no. 3, pp. 770–783, July 1997.
12. J. Lowell, "Military Applications of Localization, Tracking, and Targeting," *IEEE Wireless Communications*, vol. 18, no. 2, pp. 60–65, 2011.
13. S. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*, Prentice Hall, Upper Saddle River, NJ, 1993.
14. Y. Chan and K. Ho, "A Simple and Efficient Estimator for Hyperbolic Location," *IEEE Transactions on Signal Processing*, vol. 42, no. 8, pp. 1905–1915, 1994.
15. B. Yang and J. Scheuing, "Cramer-Rao Bound and Optimum Sensor Array for Source Localization from Time Differences of Arrival," *IEEE International Acoustics, Speech, and Signal Processing*, vol. 4, pp. 961–964, 2005.
16. Y. Bar-Shalom, X. Rong Li, and T. Kirubarajan, *Estimation with Application to Tracking and Navigation*, John Wiley Sons, New York, NY, 2001.
17. B. Call, R. Beard, C. Taylor, and B. Barber, "Obstacle Avoidance For Unmanned Air Vehicles Using Image Feature Tracking," *AIAA Guidance, Navigation, and Control Conference*, pp. 3406–3414, 2006.

18. R. Hartley and A. Zisserman, “Multiple View Geometry in Computer Vision,” *Cambridge University Press*, vol. 23, 2005.
19. T.M. Clemons and Kuo-Chu Chang, “Sensor Calibration using In-Situ Celestial Observations to Estimate Bias in Space-Based Missile Tracking,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 2, pp. 1403–1427, Apr 2012.
20. D. Unsal and K. Demirbas, “Estimation of Deterministic and Stochastic IMU Error Parameters,” in *2012 IEEE/ION Position Location and Navigation Symposium (PLANS)*, Apr 2012, pp. 862–868.
21. “IEEE Standard Specification Format Guide and Test Procedure for Linear, Single-Axis, Non-Gyroscopic Accelerometers,” *IEEE Std 1293-1998 (R2008)*, pp. 1–249, Jul 2011.
22. Y. Kosuge and T. Okada, “Statistical Analysis For Radar Bias Error Estimation in a Data Fusion System of 3-Dimensional Radars,” *IEEE Industrial Electronics Society*, vol. 3, pp. 2001–2006, 2000.
23. A. Rafati, B. Moshiri, K. Salahshoor, and M. Tabatabaei pour, “Asynchronous Sensor Bias Estimation in Multisensor-Multitarget Systems,” in *2006 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Sep 2006, pp. 402–407.
24. X. Lin, Y. Bar-Shalom, and T. Kirubarajan, “Multisensor Multitarget Bias Estimation for General Asynchronous Sensors,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 3, pp. 899–921, Jul 2005.
25. C.C. Liebe, “Star Trackers for Attitude Determination,” *Aerospace and Electronic Systems Magazine, IEEE*, vol. 10, no. 6, pp. 10–16, Jun 1995.

26. “IEEE Standard for Terminology and Test Methods for Analog-to-Digital Converters,” *IEEE Std 1241-2010 (Revision of IEEE Std 1241-2000)*, pp. 1–139, Jan 2011.
27. A. Wu, “SBIRS High Payload LOS Attitude Determination and Calibration,” in *1998 IEEE Aerospace Conference*, Mar 1998, vol. 5, pp. 243–253 vol.5.
28. R.O. Green, J.E. Conel, J. van den Bosch, M. Shimada, and M. Nakai, “On-Orbit Calibration of the Japanese Earth Resources Satellite-1 Optical Sensor using the Airborne Visible-Infrared Imaging Spectrometer,” in *Geoscience and Remote Sensing Symposium, 1993. IGARSS '93. Better Understanding of Earth Environment., International*, Aug 1993, pp. 1312–1314 vol.3.
29. S. Julier and J. Uhlmann, “A Non-Divergent Estimation Algorithm in the Presence of Unknown Correlations,” *American Control Conference*, 1997.
30. S. Julier and J. Uhlmann, “Simultaneous localisation and map building using split covariance intersection,” *IEEE Intelligent Robots and Systems*, vol. 3, pp. 1257–1262, 2001.
31. K. Novak, “Numerical Analysis for Scientific Computing,” Class Handout, MATH 674. Graduate School of Engineering and Management, Air Force Institute of Technology, Wright-Patterson AFB OH, June 2008.
32. N.M. Dempster and D.B. Rubin, “Maximum Likelihood from Incomplete Data via the EM Algorithm,” *Journal of the Royal Statistical Society*, vol. 39, pp. 1–38, 1977.

<b>REPORT DOCUMENTATION PAGE</b>					<i>Form Approved</i> <b>OMB No. 0704-0188</b>	
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. <b>PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.</b>						
<b>1. REPORT DATE</b> (DD-MM-YYYY) 26-03-2015		<b>2. REPORT TYPE</b> Master's Thesis			<b>3. DATES COVERED</b> (From — To) Sept 2013 — Mar 2015	
<b>4. TITLE AND SUBTITLE</b>  Non-Linear Optimization Applied to Angle-of-Arrival Satellite-Based Geolocation with Correlated Measurements					<b>5a. CONTRACT NUMBER</b>  <b>5b. GRANT NUMBER</b>  <b>5c. PROGRAM ELEMENT NUMBER</b>  <b>5d. PROJECT NUMBER</b>  <b>5e. TASK NUMBER</b>  <b>5f. WORK UNIT NUMBER</b>	
<b>6. AUTHOR(S)</b>  Sprang, Joshua S., 2d Lt, USAF					<b>8. PERFORMING ORGANIZATION REPORT NUMBER</b>  AFIT-ENG-MS-15-M-044	
<b>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</b> Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way WPAFB OH 45433-7765					<b>10. SPONSOR/MONITOR'S ACRONYM(S)</b>  <b>11. SPONSOR/MONITOR'S REPORT NUMBER(S)</b>	
<b>9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)</b>  INTENTIONALLY LEFT BLANK					<b>12. DISTRIBUTION / AVAILABILITY STATEMENT</b>  DISTRIBUTION STATEMENT A: APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.	
<b>13. SUPPLEMENTARY NOTES</b>  This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.						
<b>14. ABSTRACT</b> A common remote sensing application is producing geolocation estimates for an object of interest from multiple sensor platforms. Geolocation estimates are desired to help improve situational awareness when dealing with space objects that do not actively broadcast their location. A depiction of the error parameters are calculated in conjunction with the positional estimates. Problems occur when multiple measurements from a single sensor are used to estimate a location due to correlations in sensor error. A non-linear optimization approach is presented for determining geolocation estimates and their associated error parameters. The error parameters directly reflect the error present on the individual measurements used to produce the position estimates. Correlations in errors are dealt with by augmenting the non-linear optimization with a covariance intersection algorithm. Finally, the ability to account for correlated errors within the optimization algorithm is analyzed using Monte-Carlo simulations. The ability to describe an objects location with a given confidence helps aid in the analysis of the system at large.						
<b>15. SUBJECT TERMS</b>  Localization, Data Fusion, Remote Sensing, Angle-of-Arrival, Non-Linear Optimization						
<b>16. SECURITY CLASSIFICATION OF:</b>			<b>17. LIMITATION OF ABSTRACT</b>		<b>18. NUMBER OF PAGES</b>	
a. REPORT	b. ABSTRACT	c. THIS PAGE	UU		95	
U	U	U	<b>19a. NAME OF RESPONSIBLE PERSON</b> Dr. A. Terzuoli , AFIT/ENG			
						<b>19b. TELEPHONE NUMBER</b> (include area code) (937) 255-3636, x4717; andrew.terzuoli@afit.edu